UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Uroš Simčič

Uvajanje metodologije Scrum v majhno podjetje za razvoj spletnih aplikacij

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

Ljubljana, 2016

UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Uroš Simčič

Uvajanje metodologije Scrum v majhno podjetje za razvoj spletnih aplikacij

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: prof. dr. Viljan Mahnič

Ljubljana, 2016

Rezultati diplomskega dela so intelektualna lastnina avtorja. Za objavljanje ali izkoriščanje rezultatov diplomskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Uvajanje metodologije Scrum v majhno podjetje za razvoj spletnih aplikacij

Tematika naloge:

Proučite značilnosti metodologije Scrum in njeno primernost za vodenje projektov v majhnih podjetjih za razvoj programske opreme. Na praktičnem primeru pokažite, kako je potekala uvedba te metodologije v podjetju, kjer delate, in analizirajte pridobljene izkušnje. Posebno pozornost namenite stopnji sprejetja posameznih praks in faktorjem, ki vplivajo na sprejemljivost Scruma kot inovacije. Pri tem se oprite na Kwon-Zmud-Cooperjev model stopenj sprejemanja tehnoloških inovacij in Rogersovo teorijo difuzije inovacij. Rezultate primerjajte s hipotezami, ki sta jih glede dolgoročne sprejemljivosti Scruma postavila Overhage in Schlauderer.

Iskreno se zahvaljujem mentorju prof. dr. Viljanu Mahniču za vso pomoč, usmerjanje in potrpežljivost. Zahvaljujem se tudi vsem ostalim, ki so mi pomagali na moji študijski poti.

V opomin sebi.
Bolečina ni nič v primerjavi z zmago,
osvojeno po bolečini.

Kazalo

Povzetek

Naslov: Uvajanje metodologije Scrum v majhno podjetje za razvoj spletnih aplikacij

Diplomsko delo opisuje uvajanje metodologije Scrum v manjše podjetje, ki se ukvarja z razvojem spletnih strani in aplikacij, ter analizira, do katere stopnje je podjetje usvojilo predpisane prakse in kateri faktorji so imeli na to največji vpliv.

V uvodnih poglavjih je teoretično predstavljeno splošno vodenje projektov in vodenje projektov z agilno metodologijo Scrum.

Sledi praktični primer uvajanja metodologije Scrum, kjer so na projektu dodelave spletne aplikacije opisani uporabljeni procesi metode Scrum in argumentirana vsa odstopanja od teorije glede na poslovne in razvijalske procese podjetja.

V zadnjem delu je analizirana uvedba metode v podjetje s teorijo difuzije inovacij, s katero je bilo preverjeno, do katere mere je podjetje sprejelo posamezne prakse, kateri faktorji so imeli na sprejetje največji vpliv in kako se mnenja razvijalcev ujemajo s hipotezami o dolgoročni sprejemljivosti Scruma, ki sta jih postavila Overhage in Schlauderer.

Ključne besede: Scrum, agilne metodologije, vodenje projektov, analiza uvedbe metodologije Scrum, difuzija inovacij

Abstract

Title: Introducing Scrum into a small web application development company

The thesis describes the introduction of the Scrum agile methodology into a small company that deals with developing websites and web applications. It also analyzes the level of assimilation of the methodology and looks into the factors with the greatest influence on its acceptance.

General project management and project management with the Scrum agile methodology are described in the introductory chapters.

The next section shows a practical example of developing an additional functionality for a web application where Scrum methodology processes are used and described. Moreover, any deviation from the theory is justified with regard to the company's business and development processes.

The last section analyzes the acceptance of Scrum with the theory of diffusion of innovation to examine the extent of accepting individual methodology practices, the most important factors for their acceptance and how opinions of the developers compare with long-term acceptance of the Scrum hypotheses set by Overhage and Schlauderer.

Keywords: Scrum, agile methodology, project management, Scrum acceptance analysis, diffusion of innovation

Poglavje 1 Uvod

Z razcvetom interneta in vedno bolj popularnih mobilnih naprav, kot so pametni mobilni telefoni in tablični računalniki, se je v zadnjem času močno povečalo povpraševanje po razvoju tako mobilnih kot spletnih aplikacij in spletnih strani, kar je vzpodbudilo rast ter nastanek številnih podjetij. Zaradi rasti podjetja in povečanja kadra se pojavi potreba po vpeljavi metodologije, s katero lahko podjetje še naprej dosega optimalne rezultate. V praksi se za vodenje projektov uveljavljajo agilne metode [5], ki so še posebej primerne takrat, ko se zahteve naročnika hitro spreminjajo. Spremembe v zahtevah so še posebej pogoste pri razvoju spletnih strani in aplikacij, kjer lahko naročnik šele takrat, ko dobi v roke neki oprijemljiv rezultat, formulira, kaj dejansko potrebuje za končni produkt. Raziskava organizacije VersionOne iz leta 2008 [6] kaže zelo pozitivne rezultate izboljšanja produktivnosti, zmanjšanja stroškov in izboljšanja kvalitete dela v podjetjih, ki so uvedla metodologijo Scrum.

Avtor diplomskega dela je zaposlen kot vodja projektov v manjšem podjetju, ki se ukvarja z elektronskim poslovanjem, večinoma pa z razvojem spletnih strani in aplikacij. V zadnjih letih se je z rastjo podjetja pojavila potreba po uvedbi metodologije, zato je podjetje sprožilo uvajanje metodologije Scrum. V diplomskem delu je tako prikazano uvajanje metodologije Scrum za potrebe podjetja in je tudi analizirano, do kakšne mere jo je podjetje uspešno uvedlo.

V diplomskem delu je v uvodnih poglavjih predstavljeno splošno vodenje projektov in vodenje projektov z agilnimi metodologijami, natančneje – z metodo Scrum, kjer so predstavljeni in obrazloženi vsi njeni procesi. Sledi poglavje s praktičnim primerom uvajanja metodologije v podjetju, kjer je avtor zaposlen, in so opisani vsi uporabljeni procesi izvajanja projekta ter njihova utemeljena odstopanja od načel metode Scrum. Na koncu analiziramo uvedbo metode v podjetje s pomočjo teorije difuzije inovacij, kjer prikažemo, do kakšne mere je podjetje sprejelo metodo, kateri so najpomembnejši faktorji za njeno sprejetje in kako se mnenja razvijalcev ujemajo s hipotezami o dolgoročni sprejemljivosti Scruma, ki sta jih postavila Overhage in Schlauderer.

Poglavje 2 Vodenje projektov

Skladno s pravili in pojmi knjige PMBOK® Guide [18] v tem poglavju povzemamo temeljne in najpogostejše procese, ki jih v podjetju izvaja in se z njimi srečuje vodja projektov. Procesi se glede na metodologijo dela v podjetju lahko izvajajo na drugačne načine in v drugačnem zaporedju, vendar so v osnovi enaki in nujno potrebni za načrtovanje, nadzor in zaključevanje vsakega projekta.

2.1 Kaj je projekt

Projekt je začasna namera posameznika ali skupine ljudi, da ustvarijo edinstven izdelek, storitev ali rezultat. Časovno omejena narava projekta kaže, da ima projekt začetek in konec, med njima pa se izvajajo posamezne aktivnosti, ki privedejo do vnaprej zastavljenega cilja. Projekt se konča, ko so doseženi njegovi cilji, ali pa je ukinjen. Rezultat projekta je lahko nekaj oprijemljivega ali neoprijemljivega. Čeprav so lahko v projektu prisotni ponavljajoči se elementi in aktivnosti, to ponavljanje ne spremeni temeljne, edinstvene značilnosti posameznega projektnega dela. Tako je lahko vsaka spletna stran ali aplikacija zgrajena s podobnimi ali enakimi orodji, izdelana s strani enakih ali različnih ekip ljudi, vendar bo na koncu edinstvena glede na unikatne potrebe vsakega naročnika.

2.2 Projektno vodenje

Projektno vodenje je kombinacija uporabe znanja, veščin, orodij in tehnik pri posameznih aktivnostih za doseganje ciljev projekta. Projektno vodenje se doseže z ustrezno uporabo in integracijo petih skupin procesov vodenja projektov: inicializacija, planiranje, izvajanje, kontrola in nadzor ter zaključek.

Osnovne naloge vodenja projekta so:

- prepoznavanje potreb projekta,
- obravnavanje različnih potreb, skrbi in pričakovanj pri načrtovanju in izvajanju,
- vzpostavitev, vzdrževanje in izvajanje komunikacije med sodelujočimi,

- upravljanje vseh sodelujočih pri doseganju projektnih zahtev in ustvarjanju rezultatov,
- uravnoteženje projektnih omejitev, kot so obseg, kvaliteta, urnik, sredstva, viri in tveganja.

Razmerje projektnih omejitev je tako, da če se spremeni ena od omejitev, se bo verjetno spremenila tudi katerakoli druga omejitev. Na primer, če se končni datum projekta skrajša, je verjetno treba povečati sredstva, da se lahko na projekt doda nove vire in s tem dokonča enako količino dela v krajšem času. Če povečanje sredstev ni mogoče, verjetno sledi zmanjšanje obsega ali kvalitete. Pri izvajanju projekta vedno obstaja možnost za spremembe, zato je razvoj načrta vodenja projekta iterativna dejavnost v celotnem življenjskem ciklu projekta. Iterativna dejavnost strmi h konstantnemu izboljšanju in natančnejšemu načrtu vodenja projekta takoj, ko so na voljo nove in podrobnejše informacije. To omogoča vodji projektov, da bolje oceni zalogo dela in ga upravlja z večjo natančnostjo med izvajanjem. Ker so pri razvoju spletnih strani spremembe pogoste, je potrebno stalno prilagajanje načrta, sicer lahko projekt zaide v težave in postane neobvladljiv.

2.3 Vloge in vplivi na projektu

Projektni vodja je oseba, določena s strani organizacije, katera vodi ekipo, odgovorno za doseganje ciljev projekta. **Sponzor** je oseba ali skupina, znotraj ali zunaj organizacije, ki priskrbi sredstva in podporo projektu. **Naročnik** je oseba ali organizacija, ki bo projekt potrdila in postavila vsa pričakovanja in zahteve projekta. **Razvojna skupina** vsebuje vodjo projekta in posameznike, ki sodelujejo in izvajajo delo projekta za dosego njegovih ciljev. **Deležnik** (ang. **Stakeholder**) je posameznik, skupina ali organizacija, ki lahko vpliva na ali je vplivana s strani odločitve, aktivnosti ali rezultata projekta.

2.4 Procesi vodenja projektov

Proces je skupina medsebojno povezanih dejanj in aktivnosti za doseganje končnega cilja v obliki produkta, storitve ali rezultata. Procesi projekta so izvedeni s strani projektne ekipe s sodelovanjem deležnikov in načeloma spadajo v dve veliki kategoriji. Prva so **procesi vodenja projektov**, kateri zagotavljajo efektiven potek projekta v njegovem življenjskem ciklu. Druga kategorija pa so **produktno orientirani procesi**, kateri definirajo in izdelajo produkt projekta. Procesi iz obeh skupin se med projektom prekrivajo in sodelujejo med seboj. Kot primer, obseg projekta ne more biti definiran, če sami cilji še niso v celoti zastavljeni, kar je pogost pojav pri

razvoju spletnih strani. Agilen pristop k razvoju omogoča, da se projekt začne izvajati kljub tej neznanki in tako omogoča dobro prekrivanje procesov iz obeh kategorij.

Poglavje 3 Agilne metodologije

Metodologijo bi najbolje opisali kot skupek vseh procesov, katere izvajamo pri doseganju cilja ali želenega rezultata [9]. Zajema vse procese: od tistih, povezanih z načrtovanjem projekta, interne in zunanje komunikacije ter pravila odločanja sodelujočih na projektu do samega izvajanja nalog. Metodologija ima formalne in neformalne elemente pri izvajanju posameznih aktivnosti. Razlika med njimi je v dokumentaciji posameznih korakov, procesov, pravil ali standardov pri izvajanju aktivnosti.

3.1 Prisotnost metodologij v manjših podjetjih

Neformalne metodologije so občutno bolj prisotne v manjših podjetjih, kjer velik del znanja opravljanja posameznih nalog ni dokumentiran, ampak je prepuščen znanju posameznikov. Tako tudi v našem podjetju ni bilo velike potrebe po uveljavi formalne metodologije za večino ali vse procese, predvsem zaradi manjšega števila zaposlenih, kateri so lahko z uporabo neformalnih elementov dobro in učinkovito izvajali svoje aktivnosti. V okviru manjše razvojne skupine znotraj enega poslovnega prostora lahko posamezniki verbalno izmenjajo vse ključne informacije in tako časovno hitreje izvedejo aktivnost, v primerjavi s sledenjem uradnega procesa pridobivanja informacij, kateri lahko traja dlje časa. V našem podjetju zaradi povečanja razvojne ekipe, ki se je porazdelila v različne poslovne prostore, neformalna metodologija ni več zadostovala, zahtevana raven komunikacije v okviru procesov je namreč postala veliko višja zaradi preprečevanja izgube informacij. Zaradi rasti podjetja se je tako pojavila potreba po vpeljavi formalne metodologije, s ciljem, da bi podjetje časovno in stroškovno strmelo k optimalnemu doseganju ciljev. Odločili smo se za metodologijo Scrum, ki je s svojimi lastnostmi takrat najbolj ustrezala delovnim procesom podjetja. Ker je uradna metodologija dokumentirana, sej je pokazalo, da tudi omogoča lažje uvajanje novega kadra, boljše razumevanje in dobro izhodiščno točko za nadaljnjo optimizacijo.

3.2 Agilne metodologije

Uspešno izvajanje nekaterih aktivnosti pri razvoju spletnih strani in aplikacij zahteva več kot eno osebo, bodisi zaradi kvantitete ali zapletenosti dela. Zato se aktivnosti loti razvojna skupina

in ne samo posameznik, kar pa zahteva večjo mero koordinacije in organiziranosti. Dejavniki, kot so stroški, čas in obseg projekta morajo biti nenehno pod nadzorom. Za razvoj takih projektov so najprimernejše agilne metodologije [5] (angl. Agile). V razvoju spletnih strani se cilji projekta velikokrat spremenijo, zato je malo verjetno, da bo končni rezultat popolnoma tak, kot si ga je naročnik zamislil na začetku. Spremembe na katerikoli stopnji projekta so za agilne metodologije dobrodošle in odlično vplivajo tako na cilje projekta kot na zadovoljstvo naročnika.

Agilne metodologije so skupina iterativnih metod, ki temeljijo na iterativnih razvojnih procesih [26]. Vsaka iteracija je majhen samostojen projekt z aktivnostmi, kot so analiza zahtev, načrtovanje, implementacija, testiranje in potrditev naročnika, ter je relativno kratka in strmi k izdaji delujočega inkrementa celotne funkcionalnosti. Naročnik na podlagi inkrementa, in ne špekulacij, pred začetkom projekta opredeli oz. prilagodi svoje zahteve, katere se uporabijo za planiranje vseh prihodnjih iteracij. Delujoči inkrementi v sklopu krajših iteracij zmanjšajo nekonsistentnost razvoja programske opreme in s tem povečajo predvidljivost in učinkovitost.

Med agilnimi metodologijami najdemo tudi najbolj razširjeno Scrum ter ostale: Extreme Programming (XP), Crystal, Dynamic Systems Development Method (DSDM), Lean Development in Feature-Driven Development (FDD). Vsaka od njih ima edinstvene lastnosti in procese, vendar si med seboj delijo osnovno filozofijo in vrednote. Vse so usmerjene v iterativno dostavljanje krajših oziroma manjših, naročniku “oprijemljivih” produktov in stalno obravnavanje povratnih informacij za doseganje projektnega cilja. Poudarjajo upoštevanje sprememb med potekom projekta v nezanesljivem okolju.

3.2.1 Tradicionalni pristopi

Tradicionalni pristopi razvoja programske opreme, med drugim Waterfall [25], se izvajajo v zaporednih individualnih fazah: zajem zahtev, analiza in načrtovanje, izvedba zahtev in implementacija oziroma namestitev na produkcijo. Vrstni red faz se praviloma ne sme spremeniti, zato je vračanje na prejšnje faze izvajanja projekta v primeru sprememb nepriporočljivo in časovno zahtevno. To predstavlja največjo slabost pristopa, še posebej, če se spremembe s strani naročnika pojavijo v zadnjih fazah projekta. Prednost takega pristopa je v večjih projektih, ki lahko trajajo tudi več let, kjer so zahteve dobro načrtovane z obsežno in natančno dokumentacijo, kar je pri izdelavi spletnih strani redkost, zato se vedno bolj uveljavljajo agilne metodologije.

3.3 Zgodovina agilnih metodologij

Pojav agilnih metodologij**Error! Reference source not found.** sega v devetdeseta leta rejšnjega stoletja. Oblikovane so bile znotraj industrije razvoja programske opreme kot alternativa tradicionalnemu pristopu, ki zahteva, da je projekt v celoti vnaprej načrtovan in izveden zaporedno ter kumulativno. Tudi prvi pobudniki so bili vsi po poklicu razvijalci programske opreme, zato ima metodologija močne povezave z IT-sektorjem, čeprav se lahko uporabi v katerikoli industriji. Leta 2001, v ameriškem mestu Snowbird, je 17 profesionalcev s področja razvoja programske opreme, na podlagi vseh njihovih izkušenj in želj po alternativnem pristopu, izoblikovalo **Manifest agilnega razvoja programske opreme** [1]. V manifestu so izpostavili najpomembnejša načela agilnega razvoja programske opreme.

- Posamezniki in njihove interakcije so pomembnejše od procesov in orodij.
- Delujoča programska oprema je pomembnejša od obsežne dokumentacije.
- Sodelovanje naročnika je pomembnejše od pogodbenih pogajanj.
- Upoštevanje sprememb je pomembnejše od sledenja načrtom.

Iz osnovnih načel so pozneje izpeljali skupno 12 temeljnih smernic razvoja. Vsako od načel upošteva trenutne dinamične trende naročnikov spletnih strani in aplikacij ter je tako osnova za dobro izpeljane projekte.

3.4 Metodologija Scrum

Scrum izhaja iz predpostavke, da je razvoj programske opreme kompleksen proces, ki ga ni moč natančno načrtovati vnaprej. Je agilna metodologija, katere začetki segajo v leto 1986, in kot vse druge agilne metodologije uporablja iterativen in inkrementen pristop, kar je voda na mlin kompleksnemu razvoju programske opreme ter spletnih strani. Ker v slovenskem jeziku še ni ustaljenih prevodov terminologije metode Scrum, v oklepaju besedila navajamo angleške prevode slovenskih izrazov, povzetih po članku iz leta 2011 [14] in prikazanih v tabeli 1, razlaga izrazov in pojmov v preostanku poglavja pa je vzeta iz literature [21] in [7].

Izraz metodologije Scrum	Slovenski prevod
Daily Scrum meeting	vsakodnevni sestanek za pregled poteka del na projektu
Product Backlog	seznam zahtev (množica uporabniških zgodb)
Product Owner	produktni vodja
Release Plan	plan izdaje
Scrum Team	razvojna skupina
ScrumMaster	skrbnik metodologije
Sprint	iteracija
Sprint Backlog	seznam nalog, potrebnih za realizacijo posameznih uporabniških zgodb
Sprint planning meeting	sestanek za načrtovanje iteracije
Sprint retrospective meeting	sestanek za oceno kakovosti razvojnega procesa
Sprint review meeting	sestanek za predstavitev rezultatov iteracije
User story	uporabniška zgodba
Velocity	hitrost razvoja (število točk, ki jih razvojna skupina lahko realizira v eni iteraciji)

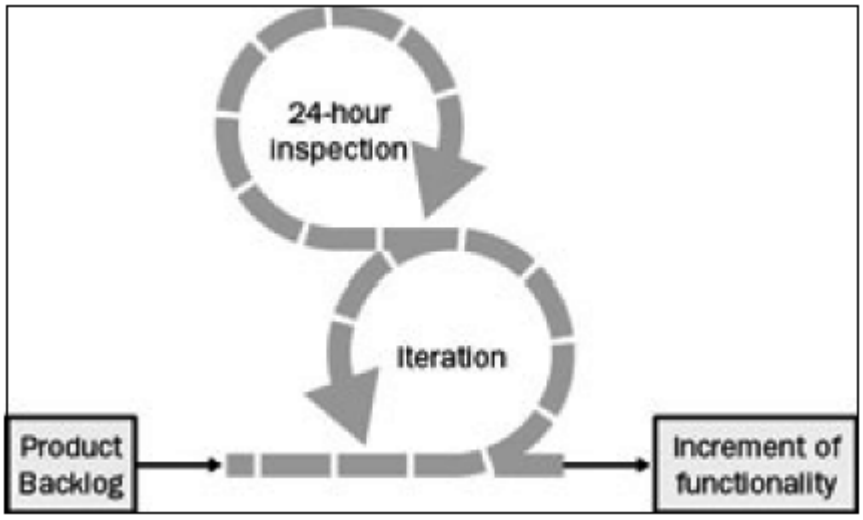
Tabela 1: Terminologija metode Scrum

3.4.1 Oris metodologije Scrum

Scrum se izvaja v iteracijah (angl. Sprint), vsako iteracijo poganja seznam nalog, potrebnih za realizacijo posameznih uporabniških zgodb, rezultat vsake iteracije pa je delujoč inkrement celotnega projekta. V teoriji iteracije trajajo od nekaj dni do približno 30 dni. Daljše trajanje iteracije pogosto ni priporočljivo, saj se začnejo izgubljati temeljne prednosti metodologije. V razvoju spletnih strani je v večini primerov funkcionalnost moč ločiti na manjše kose, zato so iteracije po navadi krajše, trajajo približno od enega do dveh tednov.

Slika 1 prikazuje potek projekta z uporabo metodologije Scrum. Pred začetkom iteracije se razvojna skupina na podlagi seznama zahtev projekta odloči, katere funkcionalnosti lahko dostavi kot delujoč produkt v časovnem okvirju ene iteracije. Projektne naloge se tako prej ali slej izvajajo v eni izmed iteracij, razen če so med izvajanjem projekta na željo naročnika odstranjene. Napredek iteracije se dnevno nadzoruje in predstavi v času vsakodnevnih, hitrih in stoječih sestankov za pregled poteka del na projektu, kar se imenuje Daily Scrum. Vsaka

iteracija dostavi potencialno delujoč izdelek, inkrement, neodvisen od celotnega cilja projekta, nekaj, kar lahko naročnik pregleda in si ustvari boljšo sliko končnega produkta.



Slika 1: Potek projekta z uporabo metodologije Scrum [21]

Projekt izdelave spletne strani ali aplikacije se začne kot vizija naročnika ali podjetja z željo po donosnosti naložb. Težko ali celo nemogoče ga je popolnoma natančno načrtovati vnaprej. Vizija in specifikacije lahko tekom iteracij ostanejo enake, ali pa se po potrebi naročnika spremenijo že med razvojem, zato je smiselno projekt izpeljati z uporabo metodologije Scrum.

3.4.2 Vloge metodologije Scrum

Scrum predpisuje tri vloge: **razvojna skupina** (angl. Scrum Team), **skrbnik metodologije** (angl. Scrum Master) in **produktni vodja** (angl. Product Owner).

Produktni vodja dobro pozna težavo naročnika in je odgovoren za vse njegove interese, donosnost naložb, zagotavlja začetna in nadaljnja sredstva za projekt in postavlja prvotne zahteve ter časovne okvirje projekta. Poleg tega je zadolžen za zahteve projekta, prikazane v **seznamu zahtev** (angl. Product Backlog). Z rednim prioritiziranjem zahtev izpostavi najpomembnejše zahteve za prvo in poznejše iteracije ter s tem omogoča razvoj pomembnejših funkcionalnosti pred ostalimi.

Razvojna skupina je samoorganizacijska entiteta, odgovorna za razvoj funkcionalnosti. Zadolžena je za samostojno, po lastni izbiri, izvajanje nalog v času posamezne iteracije. Ugotoviti in realizirati mora, kako iz seznama zahtev narediti inkrement funkcionalnosti

celotnega projekta v časovnem okvirju iteracije. Člani razvojne skupine so kolektivno odgovorni za uspeh vsake iteracije in celotnega projekta.

Skrbnik metodologije v prvi vrsti skrbi za pravilno izvajanje procesa Scrum. Vsi prisotni pri izvajanju projekta morajo poznati metodologijo in njihove naloge ter pričakovanja. Prav tako mora skrbnik poskrbeti, da se Scrum vklaplja v kulturo organizacije tako, da bo zagotavljal pričakovane koristi. Ko se metodologija uvaja v podjetje, mora redno spremljati slabe in dobre rezultate ter se nanje ustrezno odzvati.

Vsi posamezniki, ki prevzamejo te vloge, so neposredno odgovorni za izpeljavo projekta v primerjavi z drugimi osebami v organizaciji, ki imajo lahko samo interes za projekt. Pomembno je, da so vloge v metodologiji Scrum jasno določene, vedno mora biti jasno, kdo nosi določeno odgovornost pri izvajanju projekta, kdo skrbi za donosnost naložb, kdo mora transformirati težavno tehnologijo v funkcionalnost in kdo skrbi za interese naročnika. Ločitev vlog, kot nova lastnost pri uvajanja metodologije, pripomore k produktivnosti in rezultatom projekta ter ustvarja zagon.

3.4.3 Uporabniške zgodbe

Uporabniške zgodbe [7] predstavljajo način, kako so v metodologiji Scrum zapisane zahteve uporabnika. Zahteve izdelave spletne strani so komunikacijski problem, naročnik mora predati svoje zahteve tistim, ki jih bodo realizirali. Uporabniške zgodbe so orodje za komuniciranje med poslovno in razvijalsko stranjo. Razmerje komunikacije mora na obeh straneh biti enakomerno, sicer pride do situacij, ki lahko potencialno škodijo projektu. V primeru prevladujoče komunikacije s strani naročnika, le-ta premočno narekuje funkcionalnosti in časovne okvirje. Če pa ni prevladujoče komunikacije s strani naročnika, to povzroči preveč tehničnega žargona in razvijalci izgubijo možnost, da “slišijo” pomembne zahteve naročnika.

Uporabniška zgodba opiše funkcionalnost, ki bo pomembna končnemu uporabniku ali naročniku in je sestavljena iz treh sklopov:

- pisnega opisa uporabniške zgodbe,
- pogovorov o uporabniški zgodbi z namenom definiranja vseh njenih detajlov ter
- iz sprejemnih testov in dokumentacije za preverjanje uspešnega razvoja uporabniške zgodbe.

Pisni opis mora biti vedno napisan tako, da ga bo razumel tako naročnik kot vsi prisotni pri razvoju funkcionalnosti. Tehnični žargon v opisih uporabniških zgodb ni priporočljiv, saj je

večinoma preveč tehničen za naročnika pri razumevanju cilja posamezne zgodbe. Naročniku zato doprinesemo dodatno vrednost z enostavnimi, njemu razumljivimi opisi zgodb, saj zaradi njih, ob koncu iteracije ali dokončanju posamezne zgodbe, dejansko ve, kaj je bilo dokončano.

Primer opisa uporabniške zgodbe je naslednji. Uporabnik lahko na spletno stran naloži svoj življenjepis. S tehničnega stališča je to zelo površen opis in razvojni skupini ne pove veliko, zato se za vse tehnične podrobnosti ustvarijo dodatne uporabniške zgodbe, katere so primarno namenjene razvojni skupini. Bolje je imeti več zgodb kot manj, seveda do neke zdrave mere. Vsaka dobra uporabniška zgodba je neodvisna, odprta za pogajanje, ima vrednost za uporabnika in naročnika, je ocenljiva, majhna in jo je moč testirati za uspešnost oz. dokončanje.

Naročnik in ponudnik storitev sta lahko v drugem časovnem pasu, kar pomeni, da se za čas rednih dnevnih srečanj težje dogovorita. Dobro komuniciranje z naročnikom je namreč velik del uspeha projekta in mnogokrat opraviči zamude ter nepričakovane zaplete. Uporabniške zgodbe so zato odlično in nujno orodje pri uvajanju metodologije Scrum, naročniku brez tehničnega predznanja omogočajo dobro predstavo in stanje projekta samo s poročilom o dokončanosti uporabniških zgodb tekom iteracije ali projekta.

3.4.4 Seznam zahtev, množica uporabniških zgodb

Uporabniške zgodbe agregiramo v seznamu zahtev (angl. Product Backlog), kar je zbirka vseh uporabniških zgodb, v katerih je zajeta celotna funkcionalnost projekta. Zanj skrbi in jih nenehno prioritizira produktni vodja glede na potrebe naročnika, saj je seznam dinamičen in spremenljiv med izvedbo projekta. Ob začetku projekta ni potrebe po celovitem zapisu vseh predvidenih funkcionalnosti v obliki uporabniških zgodb, lahko so zapisane samo vse glavne v obliki vseh podstrani, katere so vsebovane v končni spletni strani, ali pa so lahko zapisane samo tiste za najpomembnejšo funkcionalnost. Produktni vodja na podlagi prioritet izpostavi najpomembnejše, ki se posledično umestijo v prvo iteracijo. Tekom projekta se o končnem produktu izve vedno več, zato seznam zahtev med izvajanjem projekta raste, vanj se dodajajo nove uporabniške zgodbe, katere pred začetkom vsake nove iteracije ponovno prioritizira produktni vodja.

3.4.5 Potek iteracije

Ko so definirane uporabniške zgodbe, je na vrsti načrtovanje iteracije (angl. Sprint planning meeting), ponavljajoč se proces pred vsako iteracijo med razvojem projekta. Je dvodelni dogodek, katerega se udeleži razvojna skupina, skrbnik metodologije in produktni vodja.

V prvem delu produktni vodja izpostavi najbolj prioritetne uporabniške zgodbe v seznamu zahtev, za katere želi, da bi bile realizirane v naslednji iteraciji. Razvojna skupina pa postavlja vprašanja, s pomočjo katerih si lahko ustvari jasno sliko vsake zahteve uporabniških zgodb, in pove, koliko teh zgodb lahko realizira tekom iteracije. Gre se za usklajevanje med željami produktnega vodje in zmogljivostjo razvojne skupine, rezultat pa je seznam uporabniških zgodb, ki bodo realizirane v naslednji iteraciji.

V drugem delu sestanka za načrtovanje iteracije razvojna skupina na internem posvetu razdeli uporabniške zgodbe na posamezne naloge in zahtevnost vsake naloge oceni v urah. Za vsako nalogo se zadolži eden od članov razvojne skupine. Produkt drugega dela sestanka je seznam nalog, potrebnih za realizacijo posameznih uporabniških zgodb [7] (angl. Sprint Backlog).

Po začetku iteracije skrbnik metodologije z razvojno skupino izvaja kratke vsakodnevne sestanke za pregled poteka dela na projektu (angl. Daily Scrum meeting). Vsakodnevni sestanek je kratek in jedrnat, zato tudi praviloma vsi prisotni stojijo. Razvojna skupina in skrbnik metodologije izpostavijo stanje napredka projekta, povratne informacije naročnika ali produktnega vodje in razne težave pri doseganju ciljev iteracije.

Po končani iteraciji se izvedeta dva sestanka. Prvi je sestanek za predstavitev rezultatov iteracije (angl. Sprint review meeting), kjer razvojna skupina predstavi, kaj je razvila v dokončani iteraciji, produktni vodji in vsem poljubno sodelujočim deležnikom. Drugi je sestanek za oceno kakovosti razvojnega procesa (Sprint retrospective meeting), kar je vpogled v dokončano iteracijo ali projekt, kjer se kritično obdelajo vsi dobri in slabi izidi, kar omogoča, da celotna organizacija stremi k nenehnemu optimiziranju poslovanja in razvoja. Ti sestanki so ključnega pomena pri uvajanju metodologije, saj se z njimi hitreje uvede tako procese, značilne za Scrum, kot edinstvene procese vsakega podjetja, ki jih potrebuje za svoje poslovanje. Malokatero podjetje pri uvajanju metodologije v celoti uvede popolnoma vse procese, večinoma se uvede tiste procese metodologije Scrum, ki prinašajo dobre rezultate.

3.4.6 Koncept “done”

Koncept “done” [23] izhaja iz angleške besede končano ali opravljeno in je kritičen za visokozmogljivo razvojno skupino. Koncept doda dodatno vrednost na trditev, da je zahteva znotraj seznama zahtev ali nalog popolnoma dokončana in pripravljena, da doprinese vrednost projektu.

Pri razvoju spletnih strani je od posameznikov razvojne skupine moč pogosto slišati odgovor na vprašanje, če je uporabniška zgodba ali opravilo pripravljeno za prenos na razvojno ali produkcijsko okolje: “Je dokončano, vendar manjka samo še …” Na koncept “done” gledamo

kot na krajši seznam opravil preverjanja dokončanosti [22], s katerimi posameznik razvojne skupine samostojno zagotovi, da je njegova naloga popolnoma dokončana in pripravljena za produkcijo. V našem podjetju so pri uvajanju definirali, da omenjeni seznam opravil minimalno obsega pregled kode, testiranje kakovosti funkcionalnosti in prenos kode oz. funkcionalnosti na bodisi razvojno ali produkcijsko okolje. Priporočena opravila na seznamu so še posodobljeni in izvedeni testi enot, spisana dokumentacija uporabe in potrditev produktnega vodje.

Koncept "done" je ena izmed težje oprijemljivih navad razvojne skupine pri uvajanju metodologije Scrum, zato ji je treba posvetiti veliko pozornosti. Za skrbnika metodologije in produktno vodjo končano pomeni, da je opravilo delujoče in del končnega produkta, medtem ko to za posameznika razvojne skupine lahko pomeni, da je svoj del zaključil, čeprav mora funkcionalnost še čez testiranje kakovosti, kar po konceptu "done" pomeni, da opravilo še ni končano. Pomembno je, da so pričakovanja koncepta "done" točno določena in se z njimi vsi strinjajo ter jih dosledno upoštevajo. Uspešno upoštevanje koncepta "done" tudi pozitivno vpliva na hitrost razvoja iteracije, saj se razvojna ekipa manjkrat vrača na nedokončane naloge, na katerih popravlja morebitne težave.

Za enostavno razumevanje in uvajanje so lahko razvojni ekipi v pomoč sledeča vprašanja.

- Smo razvili pravo funkcionalnost?
- Smo funkcionalnost razvili pravilno?
- Ima nova funkcionalnost dovolj vrednosti, da razreši uporabniško zgodbo in doprinese vrednosti projektu?

3.4.7 Orodja za planiranje

Agilne metode zahtevajo, da se uporabniške zgodbe v seznamu zahtev ocenijo s točkami (angl. Story Points) in da se na podlagi njih lahko izdela načrt časovnih okvirjev trajanja projekta, v agilnih metodologijah imenovan plan izdaje (angl. Release Plan). Dobra lastnost točkovnega ocenjevanja je, da glede na interni dogovor vsake organizacije ali razvojne skupine pomeni drugačen časovni okvir. Točka je abstraktna enota, lahko je časovna in predstavlja recimo idealen delovni dan brez motenj, ali pa opisuje kompleksnost oz. težavnost določene uporabniške zgodbe.

Ocene se določijo na sestanku za načrtovanje iteracije, kjer se razvojna skupina po predstavitvi uporabniških zgodb s strani produktne vodje med seboj posvetuje o pričakovani funkcionalnosti in za vsako od zgodb poda oceno v točkah. Ker je lahko ocena različna ali pristranska, lahko razvojna skupina uporabi eno od tehnik ocenjevanja zgodb. Ena izmed bolj priljubljenih,

enostavnih in hitrih je igra poker planiranja (angl. Planning poker), kjer vsak posameznik na fizični karti poda oceno zahtevnosti uporabniške zgodbe v točkah. Vse vrednosti so zakrite, dokler vsi posamezniki hkrati obrnejo karte in razkrijejo vrednosti na njih. V primeru odstopanj nastopi razprava in se postopek ponovi, dokler niso vsi prisotni s točkovno oceno zadovoljni. Manjša odstopanja v ocenah med posamezniki niso pomembna in zelo malo, če sploh, vplivajo na časovni okvir projekta, zato je večinoma bolje, da jim ne posvečamo preveč časa.

Hitrost razvoja (angl. Velocity) nam pove, koliko uporabniških točk je razvojna skupina zmožna realizirati v času ene iteracije. Hitrost razvoja ločimo na ocenjeno in dejansko. Ocenjeno določimo pred začetkom vsake iteracije, ko še ni znano, koliko točk bo razvojna skupina v praksi predelala, torej kakšna bo dejanska hitrost razvoja. Določimo jo z izmerjenimi hitrostmi razvoja iz preteklih iteracij s podobnimi uporabniškimi zgodbami, ali pa hitrost ocenimo na podlagi znanja in izkušenj. Dejansko hitrost razvoja predstavlja seštevek vseh točk uporabniških zgodb, dokončanih v času ene iteracije. Upoštevajo se samo uporabniške zgodbe, ki so v celoti končane, tj. izpolnjujejo vse pogoje, določene s konceptom “done”. Nedokončanih zgodb, pa četudi so po mnenju razvijalcev dokončane 90-odstotno, pri izračunu dejanske hitrosti ne smemo upoštevati.

3.4.8 Plan izdaje

Ocenjeno hitrost razvoja in zalogo ocenjenih točk skrbnik metodologije pretvori v realne časovne okvirje z deljenjem seštevka točk uporabniških zgodb z ocenjeno hitrostjo razvoja. Rezultat je ocenjeno število iteracij trajanja projekta, katero se uporabi za definiranje plana izdaje. Kot primer, če posamezna iteracija traja teden dni, bi projekt z 90 točkami trajal 3 iteracije, torej približno 3 tedne, če je ocenjena hitrost razvoja 30 točk na iteracijo.

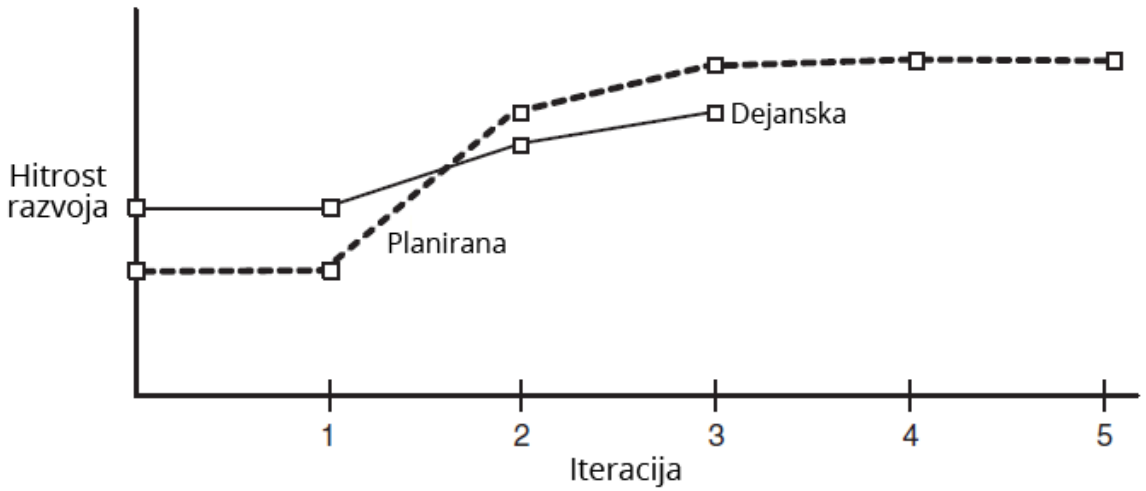
Pred izdelavo plana izdaje je treba poznati še želje časovnih okvirjev in prioritet, navezujočih se na uporabniške zgodbe naročnika ali produktne vodje, da se temu primerno lahko definirajo pričakovanja in ustrezna sredstva znotraj organizacije izvajalca. Plan izdaje je tako sestavljen iz prioritiziranih uporabniških zgodb in števila iteracij trajanja projekta. Plan se potencialno spremeni med projektom z vsakimi novimi informacijami za posamezne uporabniške zgodbe, posledično se osveži tudi vrstni red oz. prioritizacija zgodb glede na potrebe naročnika.

3.4.9 Orodja za obvladovanje poteka projekta

Dva najpomembnejša indikatorja napredka projekta sta dejanska hitrost razvoja (v primerjavi s planirano) in obseg nedokončanega dela, ki je še preostalo do konca iteracije oziroma projekta.

3.4.9.1 Hitrost razvoja

Merjenje dejanske hitrosti razvoja je enostavno, le seštejemo točke vseh, po konceptu "done" dokončanih uporabniških zgodb ob koncu iteracije. Dejansko hitrost razvoja tako primerjamo z ocenjeno hitrostjo in s tem vidimo produktivnost razvojne skupine. Slika 2 prikazuje primer planirane oz. ocenjene in dejanske hitrosti razvoja, os y, v času 3 iteracij, os x. Iz slike je razvidno, da dejanska hitrost razvoja na začetku raste in se na koncu dokaj stabilizira.



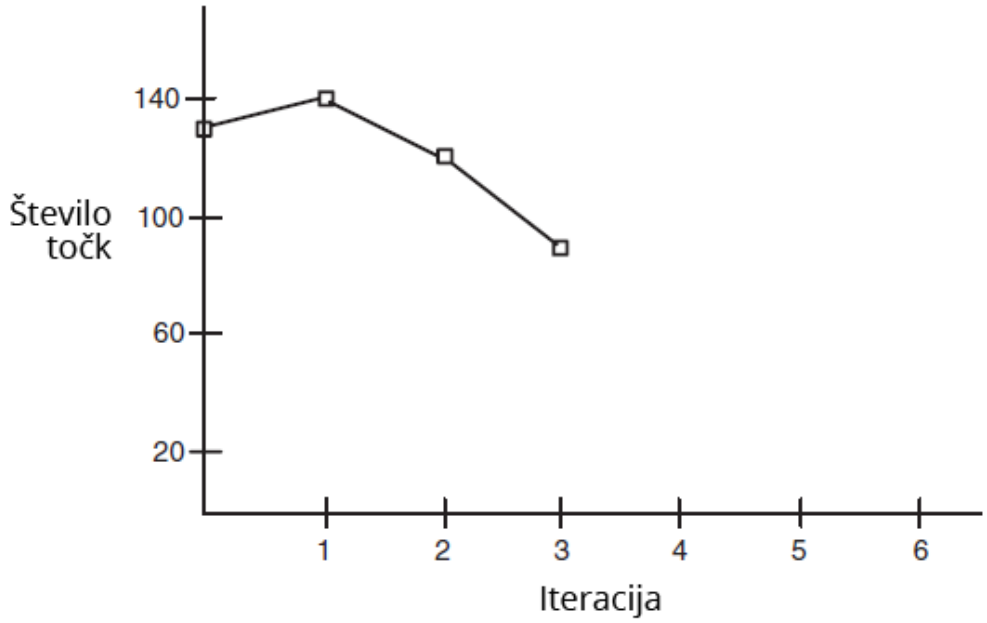
Slika 2: Primerjava planirane in dejanske hitrosti razvoja [7]

3.4.9.2 Pogoriščni diagram iteracije in projekta ali plana izdaje

Napredek je tudi moč meriti s preostankom števila ocenjenih točk vseh uporabniških zgodb med iteracijo ali celotnim projektom, kar uprizorimo na nivoju posamezne ali vseh iteracij projekta s t. i. pogoriščnim diagramom [7] (angl. Burndown chart), ki še nima ustaljenega slovenskega izraza, zato lahko v literaturi zanj najdemo tudi druga imena. Graf na sliki 3 prikazuje tako napredek plana izdaje in iteracije v obliki padajoče krivulje kot spremembe zaloge dela posameznih uporabniških zgodb v obliki strmih dvigov ali padcev krivulje. Tudi naročniku dobro prikaže vse morebitne spremembe na projektu v obliki strmih dvigov ali padcev.

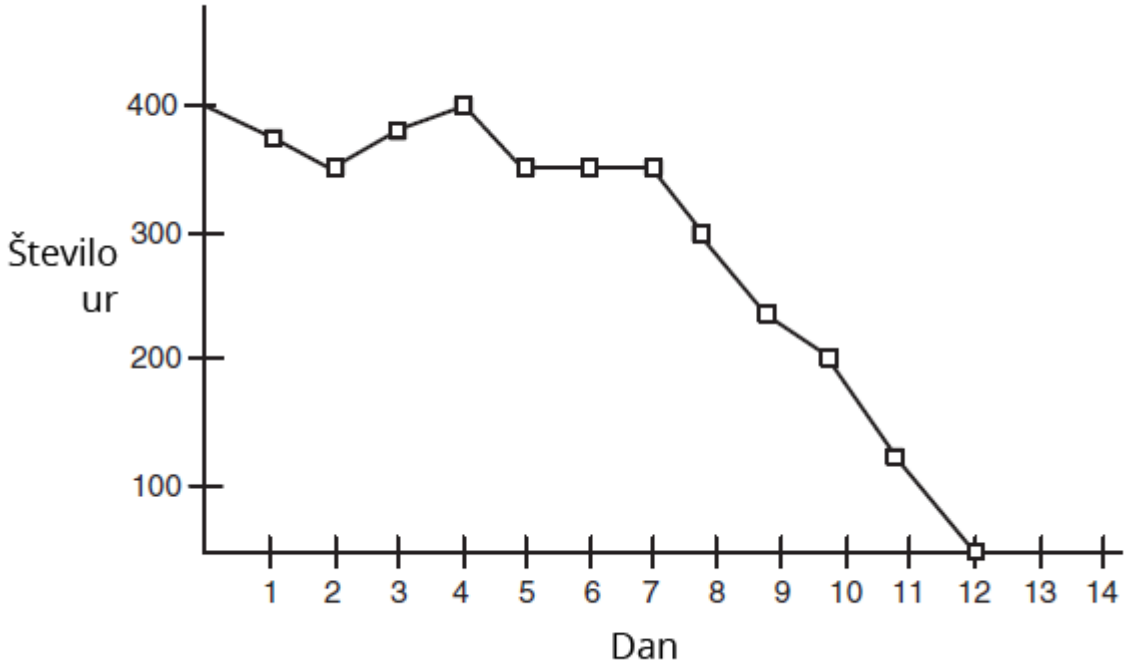
Obsežnejši pregled poteka projekta omogoča graf, ki ga prikazuje **Error! Reference source ot found.**, imenovan pogoriščni diagram projekta ali plana izdaje (angl. Release Burndown), kjer os x prikazuje iteracije plana izdaje in os y število točk vseh, še nedokončanih zgodb na začetku vsake iteracije. Na grafu lahko jasno vidimo, da je že v času prve iteracije prišlo do sprememb na projektu v obliki dviga števila zaloge točk vseh uporabniških zgodb. Pri tradicionalnem pristopu bi morali ponoviti celoten proces planiranja, medtem ko pri agilnem

pristopu v seznam zahtev (množico uporabniških zgodb) dodamo vse nove zgodbe, katere bomo ocenili pred začetkom naslednje iteracije in s tem dobili nove časovne okvirje trajanja projekta.



Slika 3: Pogoriščni diagram projekta po prvih treh iteracijah [7]

Pogoriščni diagram iteracije (angl. Sprint Burndown), prikazan na Slika 4, prikazuje napredek razreševanja nalog tekom iteracije. Os y predstavlja obseg preostalega dela, izražen v urah, ki ga izračunamo za vsak dan posebej, tako da za vsako nalogo iz Sprint Backloga ocenimo, koliko ur dela je še potrebnih za njeno dokončanje, in seštejemo vse ocene. Os x pa prikazuje časovni potek iteracije po dnevih.



Slika 4: Pogoriščni diagram iteracije projekta po 12 dnevih [7]

Poglavje 4 Primer uvajanja metodologije v podjetje

Podjetje, v katerem smo trenutno zaposleni, posluje že od leta 2000 in ima trenutno zaposlenih 12 ljudi. Podjetje nudi širok nabor storitev elektronskega poslovanja, predvsem pa oblikovanje in izgradnje spletnih mest in aplikacij. Pred približno tremi leti se je iskanje naročnikov uspešno usmerilo na trg Združenih držav Amerike, zaradi česar se je predvsem povečal kader podjetja, potreba po boljši metodi izvajanja procesov in uvedbi uradne metodologije, da smo lahko postali bolj konkurenčni na trgu.

Kmalu po usmeritvi na ameriški trg smo se povezali in postali partnerji z ameriškim podjetjem, specializiranim za spletni marketing. Podjetje ima zaposlenih približno 50 ljudi, glavno pisarno locirano v mestu San Francisco in je ravno v času našega prodora na trg potrebovalo podizvajalce pri razvoju spletnih mest in aplikacij. V našem podjetju predstavljajo naročnika, čeprav je končni naročnik drugo podjetje, ki najema našega naročnika za izvajanje storitev. Temu primerno bomo v preostanku tega poglavja v študiji primera uporabljali izraza naročnik in končni naročnik.

4.1 Priprave na uvedbo

Uvajanje metode Scrum smo izpeljali samostojno znotraj podjetja v zadnjih letih, predvsem po prodoru na ameriški trg. Vseh procesov metode nismo uvedli, saj vse lastnosti metode ne ustrezajo načinu poslovanja podjetja. Vsa odstopanja procesov od teorije metode Scrum bomo razložili in utemeljili z argumenti na koncu poglavja. Prepričani smo, da se lahko vsak proces vedno optimizira, ne trdimo, da so tisti v našem uvajanju najboljši, vseeno pa smo jih definirali z večletnimi izkušnjami poslovanja in nam doprinesejo (interno) vrednost. Uvajanje procesov metodologije smo v določeni meri tudi prilagodili glede na naše sodelovanje z naročnikom, saj nam je predstavljalo večino prihodkov.

4.1.1 Predstavitev projekta

Eden izmed produktov našega naročnika je spletna aplikacija za vodenje marketinških projektov. Skupina ljudi znotraj organizacije naročnika je zadolžena za njen razvoj in nam je v

okviru projekta predstavljala končnega naročnika, s katerim smo zaradi optimiziranja komunikacije izjemoma komunicirali neposredno. Z manjšimi dodelavami na aplikaciji smo se že srečali v preteklosti, tokrat pa je šlo za večje dodelave, v okviru načrtovanih mejnikov končnega naročnika, za prodor na trg.

Projekt je v specifikacijah naročila zajemal dodelavo večjega sklopa funkcionalnosti obstoječe spletne aplikacije in nekaj manjših neodvisnih zahtev odprave napak ter posodabljanja uporabljenih knjižnic za hitrejše delovanje aplikacije. Uporabljene tehnologije temeljijo na platformi Node.js [16] za zaledni del in Angular [2] ter React [19] za čelni del ter izrisovanje v oknu brskalnika. Večji sklop dodelave je predstavljala nova funkcionalnost grupiranja posameznih nalog v aplikaciji, prikazovanje in urejanje grup ter vsa množična urejanja nalog, ki pripadajo določeni grupi.

4.1.2 Razdelitev vlog

Vlogo skrbnika metodologije smo kot vodja projektov v podjetju prevzeli mi. Skrbeti smo morali za inicializacijo, planiranje, izvajanje, nadzor, dokončanje projekta, izvajanje dogovorjenih procesov metodologije Scrum ter vzdrževali smo redno komunikacijo s končnim naročnikom v obliki dnevnih poročil napredka in sestankov v popoldanskem času. Ker je končni naročnik v drugem časovnem pasu, se je izkazalo, da je dobra komunikacija ključnega pomena za njegovo zadovoljstvo.

Oseba na delovnem mestu vodje razvoja je prevzela vlogo produktne vodje in s tem skrbela za optimalen razvoj uporabniških zgodb in nalog pri doseganju ciljev projekta ter razvojni skupini nudila tehnično pomoč. V popoldanskem času se je udeleževala istih sestankov kot skrbnik metodologije in na njih s končnim naročnikom izmenjala vse informacije uporabniških zgodb ter tehnično odgovarjala na vsa morebitna vprašanja.

Razvojna skupina je bila sestavljena iz treh članov, in sicer iz dveh internih članov, ki sta se z razvojem aplikacije že srečala v preteklosti, in enega iz naročnikove skupine razvijalcev, kateri je bil dodeljen kot dodatna pomoč za doseganje hitrejših časovnih rokov projekta in še ni bil seznanjen z delom na aplikaciji. Eden od internih razvijalcev je bil ista oseba kot produktni vodja, saj si je s prvo vlogo zapolnil le polovico delavnika in je bil glede na svoje znanje, izkušnje ter časovne roke projekta smiselna izbira za dodatnega, polovičnega razvijalca na projektu.

4.1.3 Definiranje koncepta "done"

Potek dela posamezne naloge uporabniške zgodbe in koncept "done" sta v našem primeru tesno povezana. Pri uvajanju metodologije smo definirali 11 različnih statusov, v katerih je naloga v določenem trenutku svojega življenjskega cikla opisana v tabeli 2. S premikom naloge skozi stolpce, idealno od leve proti desni, in različne statuse v tabli Kanban [12] se izpolnijo skoraj vsi dogovorjeni pogoji koncepta "done". Opis stolpcev tabele je naslednji.

- **Seznam nalog, potrebnih za realizacijo uporabniških zgodb.** Seznam vseh nalog iteracije, na katerih se delo še ni začelo.
- **Treba je dokončati danes.** Enako kot prvi stolpec, le da lahko pričakujemo, da bodo naloge v tem stolpcu narejene isti dan, ko so bile vanj dodane. Stolpec je dodan predvsem kot prilagoditev za razreševanje vseh sprotnih zahtev drugih projektov, kar se pri nas dogaja pogosto.
- **V izvajanju.** Če se razvoj za določeno nalogo ustavi, se ji dodeli status razvoj ustavljen, sicer ima status v razvoju. Preden gre naloga v naslednji stolpec, za njo razvijalec napiše še avtomatske sprejemne teste za orodje Bamboo [3].
- **V pregledu.** Ko naloga čaka pregled napisane kode s strani drugega razvijalca, ima dodeljen status pripravljeno na pregled, v pregledu, ko je v procesu pregleda s pomočjo orodja Bitbucket [4] ali Github [10], in pregled ustavljen, v kolikor se pregled ustavi.
- **Testiranje kakovosti.** Enako kot stolpec v pregledu, le da se preveri, ali naloga ustreza oblikovni in funkcionalni rešitvi naročnika. V našem primeru izvaja to aktivnost oseba, ki je ločena od razvojne skupine in izključno zagotavlja kakovost, zato smo ji definirali samostojen stolpec.
- **Dokončano.** Naloga je pripravljena na zadnji pregled skrbnika metodologije ali produktne vodje, preden se posreduje naročniku.

Seznam nalog, potrebnih za realizacijo uporabniških zgodb	Treba je dokončati danes	V izvajanju	V pregledu	Testiranje kakovosti	Dokončano
Pripravljeno za delo	Treba je dokončati danes	V razvoju	Pripravljeno na pregled	Pripravljeno na testiranje	Dokončano
		Razvoj ustavljen	V pregledu	V testiranju	
			Pregled ustavljen	Testiranje ustavljeno	

Tabela 2: Opis vseh statusov zahtev

Vsem sodelujočim na projektu statusi omogočajo boljšo samoorganizacijo in dobro opisujejo trenutno stanje nalog. Proces se je izkazal za koristnega v primeru sprememb članov razvojne skupine med izvajanjem projekta, in če se je razvoj projekta začasno ustavil ter pozneje nadaljeval.

Zadnji pogoj koncepta “done” je pri izvajanju projekta predstavljal končni naročnik. Vse narejene uporabniške zgodbe je želel testirati v razvojnem okolju in jih potrditi, preden smo jih prenesli na produkcijsko okolje.

4.1.4 Uporabniške zgodbe

Ocena uporabniških zgodb v točkah je predstavljala abstraktno mero težavnosti in časovne ocene. Končno število točk je bil zmnožek obeh, vsaka z razponom od 1 do 8. Pričakovana težavnost z vrednostjo 1 točke je predstavljala enostavne spremembe, brez nove logike, 8 točk pa komponento z veliko neznanimi faktorji. Časovna zahtevnost zgodbe v točkah se je definirala na podoben način, 1 je predstavljala približno 1 uro dela, medtem ko jih je 8 predstavljajo približno 16 ur dela oz. dva delovna dneva. Zaradi boljšega nadzora napredka med izvajanjem projekta in manjše kompleksnosti zgodb smo se držali splošnega pravila, da so se vse zgodbe s točkovno oceno, večjo od 16, porazdelile na več manjših.

Ker smo želeli proces ocenjevanja izpeljati kar se da hitro, smo za manjša odstopanja v ocenah upoštevali povprečno vrednost, za večja odstopanja pa uporabili tehniko poker planiranja. Vsako točkovno oceno zgodbe smo še dvignili za 20 odstotkov za potrebe procesov koncepta "done", kar je pokrilo čas pregleda vsake dokončane naloge s strani drugega razvijalca in pisanje sprejemnih testov, čeprav bi morali biti le-ti po teoriji metode napisani vnaprej.

4.1.5 Planiranje projekta

Končni naročnik se je želel držati svojih interno zastavljenih rokov izida aplikacije in ji hkrati dodati čim več funkcionalnosti. Uporabniške zgodbe so se posledično izdelale, točkovno ocenile in dodelile posameznikom razvoje skupine že pred prvim sestankom za načrtovanje iteracije, saj je naročnik na podlagi prejetih ocen še spreminjal nabor uporabniških zgodb. Prvi sestanek za načrtovanje iteracije se je zato zgodil še pred začetkom projekta in ne na prvi dan razvoja projekta.

4.1.5.1 Planiranje iteracij

Ena od zahtev končnega naročnika je bila aktivno sodelovanje pri testiranju uporabniških zgodb med izvajanjem iteracije. Dogovorili smo se za dvotedenske iteracije, pri katerih se je po koncu prvega tedna, torej po petih delovnih dnevih, končane uporabniške zgodbe namestilo na testno okolje, kjer jih je lahko končni naročnik testiral in javil povratne informacije. Šesti dan se je izvedel sestanek za načrtovanje naslednje iteracije, kjer so se izdelale in časovno ocenile naloge za izbrane uporabniške zgodbe. Preostanek dni je bil namenjen razreševanju povratnih informacij končnega naročnika in prejetju končne potrditve dokončanja zgodb, da smo jih lahko zadnji, deseti, dan prestavili na produkcijsko okolje. Če v drugem tednu iteracije razvojna skupina ni bila zapolnjena s povratnimi informacijami končnega naročnika, je delala na uporabniških zgodbah naslednje iteracije.

4.1.5.2 Planiranje plana izdaje

Seštevek točk skupno 20 uporabniških zgodb pred začetkom projekta je bil 153. Iz preteklih projektov smo vedeli, da lahko v enem tednu razvoja posameznik razvojne skupine predela približno 25 točk. Ker se eden od razvijalcev s spletno aplikacijo še ni srečal in se bo moral uvesti v projekt, vendar smo vedeli, da je kar izkušen, smo njegovo hitrost razvoja zmanjšali na 20 točk. Poleg tega je eden od internih razvijalcev na projektu delal le polovico časa, zato smo njegovo hitrost ocenili na 12 točk. Skupaj smo prišli na ocenjeno hitrost 57 točk, kar je pomenilo, da bomo verjetno projekt končali v roku 3 iteracij, torej v 6 tednih.

4.1.5.3 Izbira orodja za nadzor plana izdaje

Proces izdelave plana izdaje se močno razlikuje od prvotno uporabljenega pred uvajanjem metodologije Scrum. Vodja projekta oz. sedaj lastnik metodologije se sreča z ocenjevanjem v drugačnih enotah, posledično drugačnim sledenjem napredka in z novim procesom pri poročanju časovnih okvirjev, za kar tudi potrebuje nova orodja. V našem podjetju smo se odločili za programsko opremo Jira [11], ki omogoča izvajanje projektov po metodi Scrum, in bomo iz nje v naslednjem poglavju črpali posnetke zaslonov pogoriščnih diagramov. Vse statuse nalog, omenjenih pri konceptu "done", smo definirali skupaj s funkcionalnostmi orodja in izkušnjami iz preteklih projektov.

4.2 Izvedba projekta

4.2.1 Potek iteracij

Delo na prvi iteraciji se je začelo v sredo 7. 10. 2015. Vsak dan ob 10:00 uri zjutraj smo izvedli hitri vsakodnevni sestanek za pregled poteka del na projektu, ki praviloma ni trajal več kot 15 minut. Izkazalo se je, da so se hitri sestanki na začetku projekta pogosto spremenili v sestanek s splošno vsebino, ki se ne navezuje na načela dnevnega sestanka. Ker se vsebina takih pogovorov ne navezuje na vse prisotne, je bilo treba sestanek usmeriti v pravo smer in po potrebi sklicati novega pozneje isti dan, da se je čas vseh prisotnih porabljal optimalno. Skrbnik metodologije in po potrebi produktni vodja sta z razvijalcem naročnika hitri sestanek izvajala v popoldanskem času, zato da smo bili v istem časovnem pasu kot končni naročnik.

Vsak šesti delovni dan iteracije smo namesto hitrega dnevnega sestanka izvedli sestanek za predstavitev rezultatov iteracije in za njim še sestanek za načrtovanje iteracije. Dokončane uporabniške zgodbe smo isti dan v popoldanskih urah predstavili končnemu naročniku in preostanek dni v iteraciji bodisi razreševali povratne informacije končnega naročnika ali pa delali na zgodbah iz naslednje iteracije.

Sestanek za oceno kakovosti razvojnega procesa je bil izveden vsak šesti dan iteracije v popoldanskih urah, da je bil lahko prisoten tudi naročnikov razvijalec. Glavna aktivnost sestanka je bila izmenjava informacij med celotno razvojno skupino. Izpostavilo se je vse predelane težave in posebnosti pri razvoju nalog. Naročnikovega razvijalca je bilo na začetku treba tudi malo spodbujati, da je sledil dogovorjenim procesom koncepta "done".

Vsak deseti delovni smo se s strinjanjem končnega naročnika odločili, da na koncu dneva ne naredimo posodobitve produkcijske različice aplikacije z dokončanimi uporabniškim zgodbami, saj gre za tem razvojna skupina kmalu domov, kar je lahko problematično v primeru

težav na aplikaciji. Ta proces smo zato zamaknili na vsak prvi dan naslednje iteracije v jutranje ure.

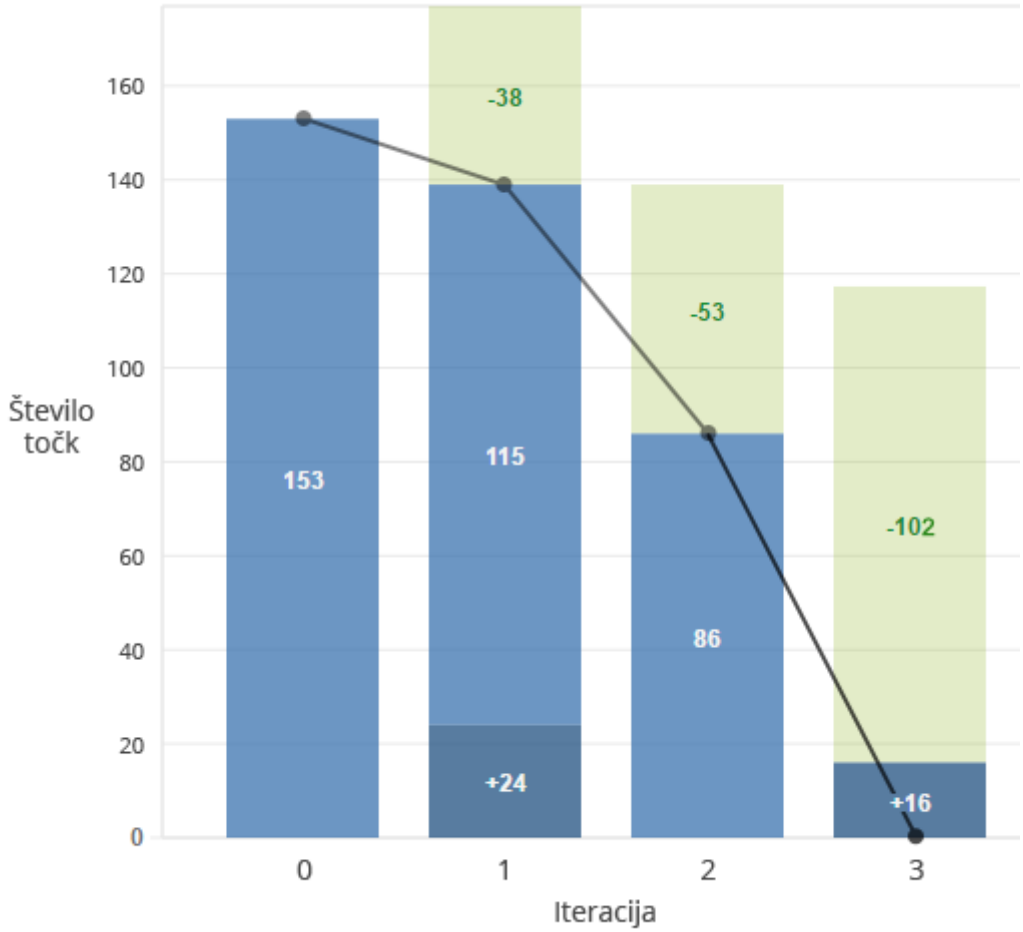
4.2.2 Potek razvoja

Med razvojem smo naleteli na nekaj posebnih situacij. Manj izstopajoča je, ko nam je končni naročnik v času prve iteracije dodal še za 24 točk dodatnih zahtev, bolj izstopajoča pa je, ko nam je končni naročnik sporočil, da želi dodelave na grupah, ki predstavljajo večji del funkcionalnosti projekta, testirati v celoti in ne postopoma. Ker je bil končni naročnik del koncepta "done" in zgodbe za funkcionalnost grup, porazdeljenih tekom treh iteracij, se je kar nekaj zgodb označilo kot dokončane šele proti koncu tretje iteracije, po potrditvi končnega naročnika. Največji indikator te odločitve je bil padec hitrosti razvoja v primerjavi z ocenjeno.

4.2.3 Nadzor projekta

4.2.3.1 Pogoriščni diagram projekta

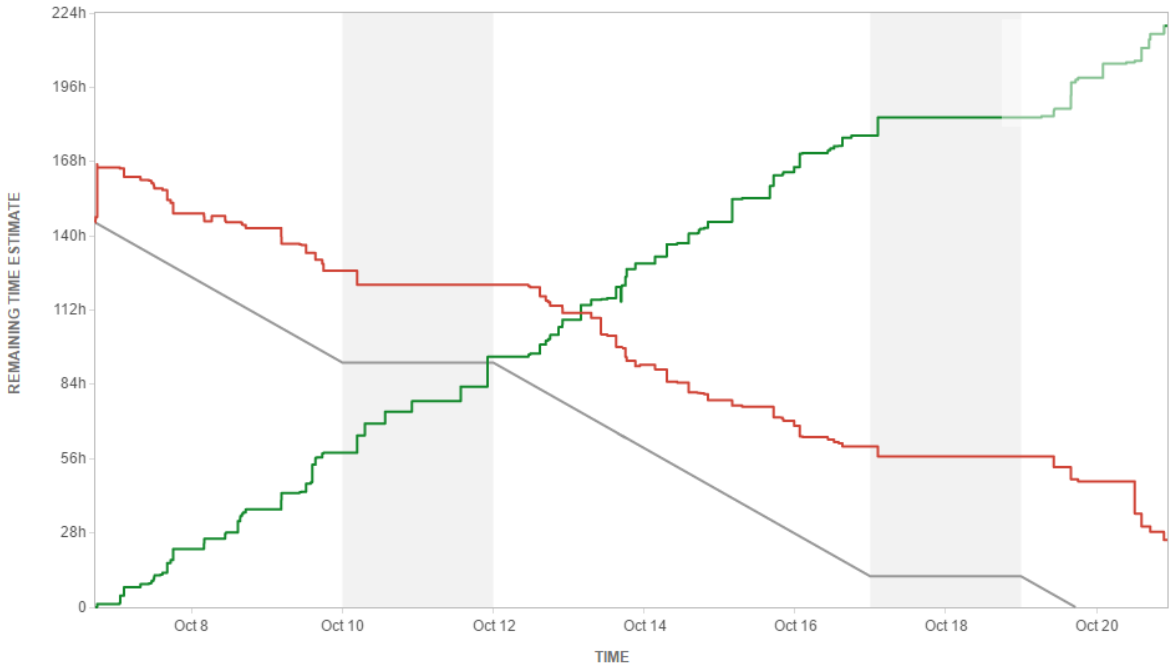
Pogoriščni diagram projekta prikazuje skupno število točk vseh nedokončanih uporabniških zgodb projekta na osi y in zaključeno iteracijo na osi x. Črna krivulja je dodana za boljši pregled preostanka skupnega števila točk na koncu vsake iteracije. Na sliki 5 je tudi prikazan seštevek točk vseh dokončanih uporabniških zgodb (negativna števila na sliki) in koliko je bilo na novo dodanih točk zaradi dodatnih uporabniških zgodb, dodanih med razvojem projekta (24 med prvo in 16 med tretjo iteracijo).



Slika 5: Pogoriščni diagram projekta tekom 3 iteracij

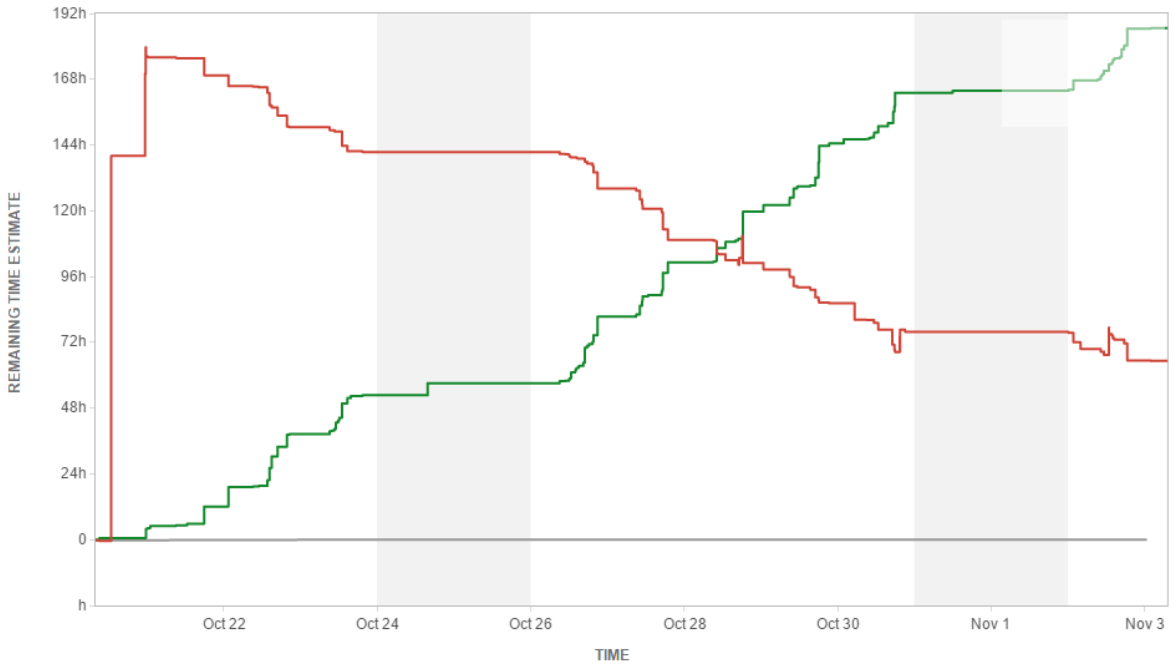
4.2.3.2 Pogoriščni diagram iteracije

Slike 6, 7 in 8 prikazujejo pogoriščne diagrame vsake od treh iteracij projekta. Rdeče krivulje prikazujejo seštevek preostalih ur vseh nalog uporabniških zgodb znotraj iteracije, zelene porabljeno število ur na projektu in sive idealen napredek zmanjšanja seštevka ur za doseganje cilja iteracije. Os y prikazuje število ur in os x dneve v iteraciji. Široki sivi pasovi prikazujejo sobote in nedelje, torej dela proste dneve, zato takrat idealen napredek (siva krivulja) ni pričakovan.



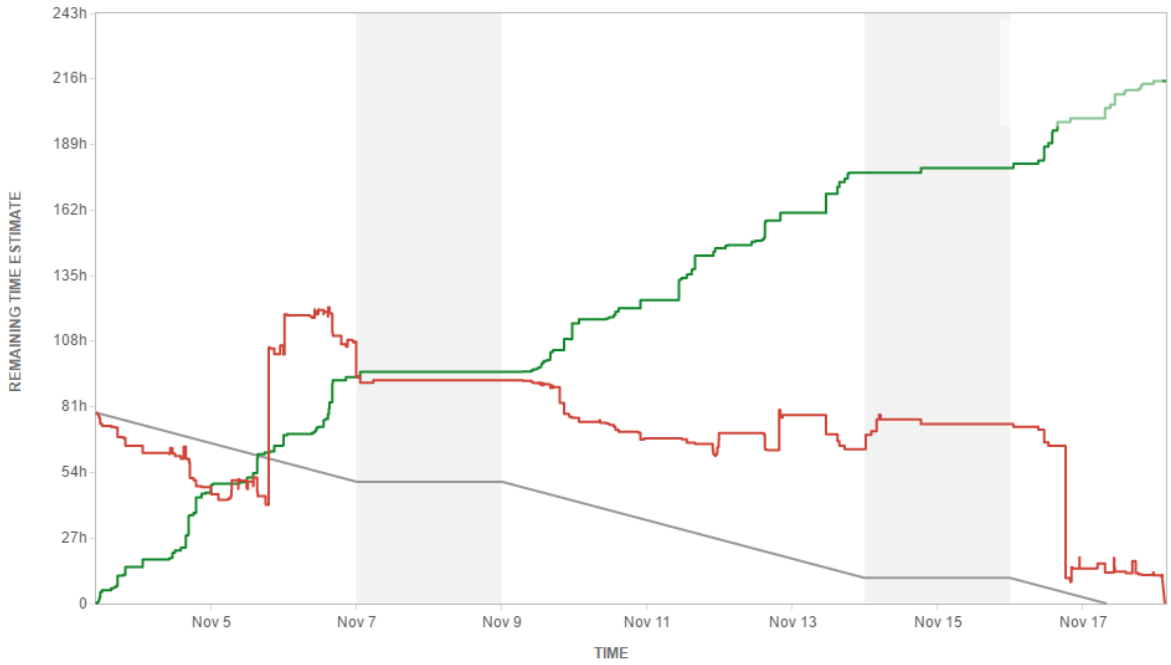
Slika 6: Pogoriščni diagram 1. iteracije

V prvi iteraciji nismo zaključili vseh načrtovanih uporabniških zgodb in na koncu smo porabili veliko več ur, kot smo jih načrtovali, predvsem zaradi povratnih informacij končnega naročnika za posamezne uporabniške zgodbe, za katere je želel bodisi dodatne funkcionalnosti ali pa rešitev ni bila v skladu z njegovimi pričakovanji.



Slika 7: Pogoriščni diagram 2. iteracije

Pogoriščni diagram druge iteracije je lep primer, ko v praksi ne gre vse tekoče. Začetni skok rdeče krivulje je posledica napačnih nastavitev v programu za vodenje projekta, prav tako se tudi siva, optimalna krivulja ni nastavila pravilno. Iteracijo smo zaradi enakih razlogov kot prvo končali z relativno velikim seštevkom ocenjenih ur nedokončanih nalog.



Slika 8: Pogoriščni diagram 3. iteracije

V tretji iteraciji smo prejeli še zadnje povratne informacije končnega naročnika, katere smo nemudoma umestili v izvedbo med iteracijo, zato so na rdeči krivulji vidni nenadni poskoki. Prav tako smo proti koncu iteracije prejeli potrdilo o dokončanju večjega sklopa funkcionalnosti grup in s tem v kratkem času zaključili večje število odprtih nalog ter zgodb, kar se vidi v nenadnem padcu rdeče krivulje malo pred 17. novembrom. Diagram tudi prikazuje večje število porabljenih ur, kot je bilo načrtovano, še posebej čisto na koncu iteracije, ko je bilo treba urediti še veliko malenkosti pred dokončnim prenosom novih funkcionalnosti na produkcijsko okolje.

4.2.4 Analiza hitrosti

V času prve iteracije smo od končnega naročnika dobili poleg dodatnih zahtev tudi največ informacij. Največji vpliv na dejansko hitrost razvoja je imelo potrjevanje dokončanja posameznih uporabniških zgodb s strani naročnika, zato ocenjene hitrosti pred drugo iteracijo nismo spreminjali in se odločili, da jo v primeru slabega napredka spremenimo pred tretjo iteracijo. Dejanska hitrost druge iteracije bila 53, vsebovala je kar nekaj praktično narejenih in hkrati nepotrjenih uporabniških zgodb funkcionalnosti grup, zato smo vedeli, da smo na dobri poti, da projekt dokončamo z enako ocenjeno hitrostjo v prvotno zastavljenih 3 iteracijah.

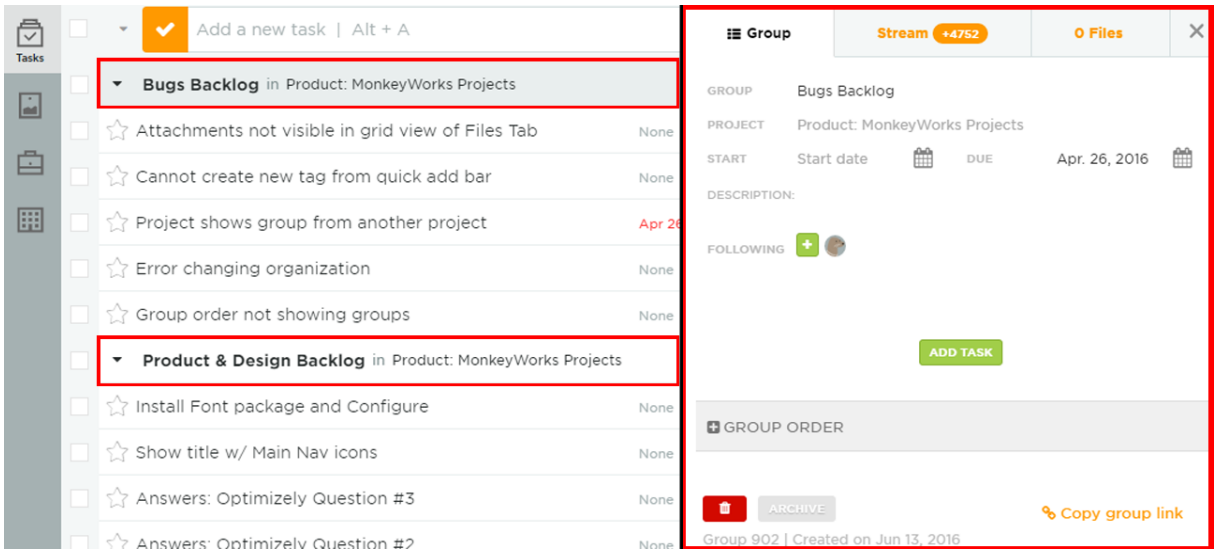
Iteracija	Ocenjena hitrost	Dejanska hitrost
1	57	38

2	57	53
3	57	102

Tabela 3: Primerjava ocenjene in dejanske hitrosti razvoja iteracij

4.2.5 Zaključek in ugotovitve

Projekt se je končal z nekaj nedokončanimi uporabniškimi zgodbami. To so bile predvsem zgodbe, ki so opisovale napake v aplikaciji, katere nismo ne mi, ne končni naročnik uspeli reproducirati tekom trajanja projekta. S strinjanjem končnega naročnika smo jih pred koncem iteracije označili kot dokončane in jih v sistemu obdržali kot evidenco, če bi se težave še kdaj ponovile. Slika 9 prikazuje končni izdelek funkcionalnosti grup s prikazom na seznamu nalog in pripadajočimi nalogami na levi ter urejevalnika posamezne grupe na desni.



Slika 9: Prikaz končnega izdelka funkcionalnosti grup

Agilna metoda nam je omogočila, da je razvoj projekta potekal nemoteno kljub pozneje dodanim zahtevam končnega naročnika, pri čemer nam procesa planiranja ni bilo treba ponoviti, le dodati smo morali nove zahteve in preveriti, kaj to pomeni za začrtane časovne roke. Končni naročnik je bil seznanjen z načinom dela po metodi Scrum, kar mu je omogočalo nenehno prioritiziranje posameznih uporabniških zgodb in je glede na napredek razvojne skupine vedel, kakšen je razvojni napredek in če lahko v projekt doda dodatne zahteve, ne da bi ogrozil končne datume.

Poleg manjših odstopanj od teorije metode Scrum sta opaznejši odstopanji še dostavljanje delujočega inkrementa funkcionalnosti in pisanje sprejemnih testov. Medtem ko smo

posamezne manjše zahteve in napake aplikacije dostavljali uspešno ob koncu vsake iteracije, je bila funkcionalnost grup razvlečena v tri iteracije, zato ob koncu prve in druge nismo dostavili delujočega inkrementa. Sprejemni testi bi morali biti definirani že pred prvim načrtovanjem iteracije, v našem primeru smo jih napisali med točkovnim ocenjevanjem zgodb.

Prav tako so se sestanki za planiranje iteracije, predstavitev rezultatov iteracije in sestanek za pregled kakovosti razvojnega procesa izvajali med samo iteracijo, dogovorjene zgodbe smo dokončali in predstavili končnemu naročniku po prvih petih dneh iteracije, potem pa delali tako na povratnih informacijah kot na zgodbah iz naslednjih iteracij. Naslednje večje odstopanje je bila vključitev končnega naročnika v koncept "done", namesto da vanj sploh ne bi bil vključen in bi dokončan ter delujoč inkrement funkcionalnosti prejel v pregled ob koncu iteracije. Taka odločitev se je sprejela predvsem zaradi boljše kontrole nad razvijanjem funkcionalnosti končnega naročnika, kar mu je omogočilo, da je potrjene funkcionalnosti izdal na produkcijsko okolje aplikacije ob koncu iteracije, sicer bi jih takrat šele prejel v predogled.

4.2.5.1 Ugotovitve iz drugih projektov pri uvajanju metode Scrum

Glavni razlogi za težave pri uvajanju metode po naših izkušnjah so v razreševanju sprotnih zahtev naročnika, katere niso povezane s trenutno izvajanim projektom, ampak s podporo in dodelavami obstoječih spletnih mest, katere je bilo v večini primerov treba razrešiti takoj v naslednjem delovnem dnevu od prejetja zahteve. Na analitičnih podatkih in grafih se to najpogosteje pozna v obliki padca hitrosti razvoja ali manjših prekinitvah izvajanja projekta. Medtem ko se take situacije lahko rešijo bodisi s planiranjem posebnih dni v iteraciji, ko se razrešujejo sprotne zahteve, ali z zadolževanjem posameznikov v razvijalni skupini za razreševanje sprotnih zahtev, je to v manjšem podjetju težko dosegljivo. Na srečo je imela razvojna skupina tega projekta zelo malo sprotnih zahtev na ostalih projektih in so se več ali manj lahko v celoti posvetili dodelavam spletne aplikacije.

Drugo najpogostejše odstopanje od teorije metode je uporaba točk pri ocenjevanju nekonsistentnih in raznovrstnih projektnih zahtev. Približno ocenjevanje trajanja projekta v točkah je nekaj, kar si pogosto ne moremo privoščiti, saj moramo že pred samim izvajanjem uporabniške zgodbe oceniti in javiti časovne ocene v urah, da lahko naročnik projekt predstavi in proda končnemu naročniku. V kolikor pride do prevelikega odstopanja od prevelike količine ocenjenih ur, obstaja tveganje, da končni naročnik projekta ne bo sprejel, premajhna ocena dela pa lahko povzroči deficit v prihodkih, zato tudi samemu planiranju posvečamo več pozornosti in časa, da lahko dostavimo bolj točne ocene v urah. Pri spremljanju napredka projekta se takrat oziramo na število preostalih ur in ne točk.

Poglavje 5 Difuzija inovacij

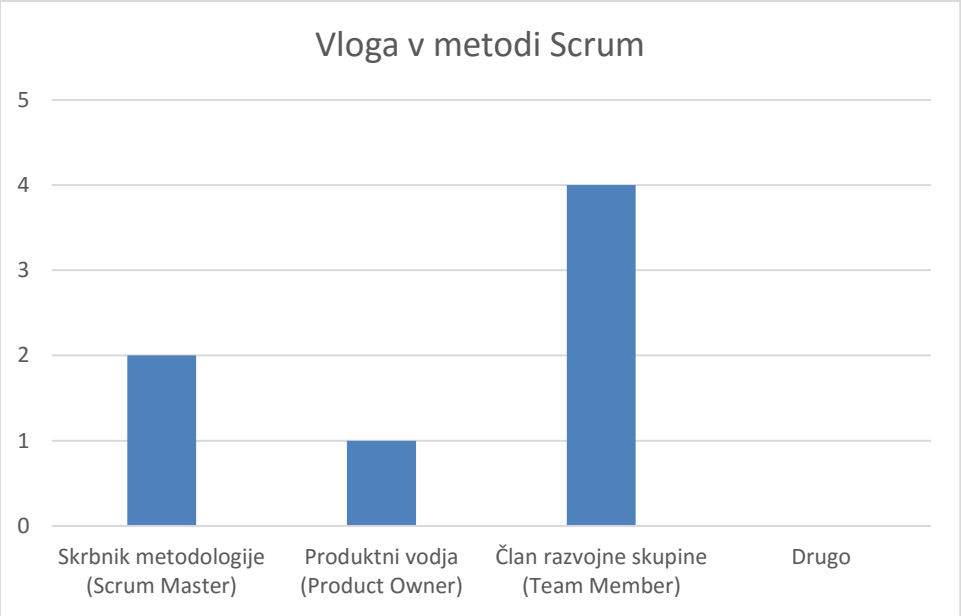
Inovacija je neki nov pojav oz. novost, katero zaznavamo kot posameznik ali organizacija. V podjetju se to kaže med drugim v obliki novih ali spremenjenih procesov poslovanja oz. drugačnega načina izvajanja dela na račun uspešnega reševanja novih težav. Difuzija inovacij je proces, pri katerem se inovacije skozi čas in čez določene kanale prenašajo do članov socialnega sistema oz. končnih uporabnikov [20]. V našem primeru je to uvajanje metodologije Scrum v podjetje. S pomočjo vprašalnika bomo ocenili, do kakšne stopnje je podjetje sprejelo metodo Scrum in kateri so faktorji, ki vplivajo na uspešno uvedbo.

5.1 Vprašalnik in rezultati

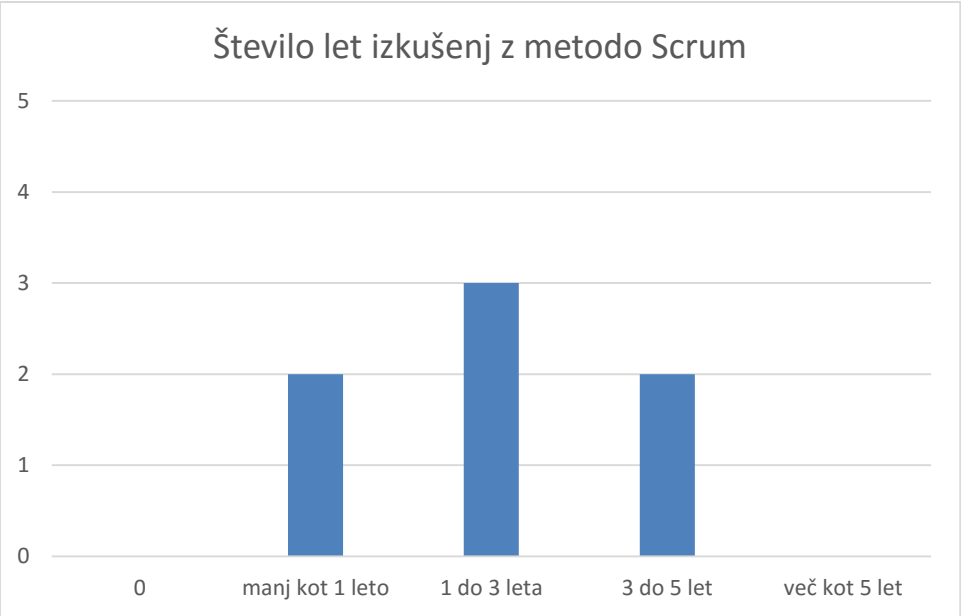
Vprašalnik, katerega je izpolnil kader v našem podjetju v mesecu maju 2016, sta definirala prof. dr. Viljan Mahnič in doc. dr. Tomaž Hovelja na Fakulteti za računalništvo in informatiko. Razdeljen je na 4 večje sklope, podrobneje opisane v preostanku poglavja. Izpolnilo ga je 7 oseb, ki se že več let ukvarjajo z izdelavo spletnih strani in aplikacij, bodisi kot razvijalci, vodje razvoja, vodje projektov ali vodje nasploh. Večina od njih je bila deležne uvedbe metode Scrum v podjetje že od samega začetka, nekateri pa so se podjetju pridružili pozneje, ko je bila metoda že do neke mere uvedena.

5.1.1 Splošni podatki o anketirancih

Prvi del vprašalnika je namenjen segmentaciji anketirancev glede na vlogo, ki jo imajo v metodologiji Scrum, in glede na obseg izkušenj z njeno uporabo. Slika 10 prikazuje porazdelitev vlog anketirancev in slika 11 število let izkušenj z metodo Scrum. Iz slike 11 je razvidno, da je za večino metoda kar nova in so jo prvič izkusili pri uvajanju v našem podjetju. V anketi prevladujejo člani razvojne skupine in mogoče je vredno omembe, da so tako skrbniki kot produktni vodja v praksi tudi lahko člani razvojne skupine na določenih projektih.



Slika 10: Razporeditev anketirancev glede na vloge v metodi Scrum



Slika 11: Število let izkušenj anketirancev

5.1.2 Stopnja sprejetja posameznih praks Scrum

Drugi del vprašalnika vsebuje vprašanja, namenjena preverjanju poznavanja posameznih elementov in procesov metode Scrum, za katere prikazujemo povprečje mediano in standardni odklon v tabeli 4. Da bi opisali proces, v katerem podjetje, skupina ali posameznik sprejme inovacijo, so Kwon, Zmud in Cooper **Error! Reference source not found.**[8], [13]] definirali

-stopenjski model sprejetja inovacije (ang. Innovation assimilation stages). Vsaka stopnja opisuje raven, do katere je inovacija prodrla pri sprejemanju:

- **iniciacija** (ang. Initiation). Povezava med inovacijo in njeno uporabo s strani skupine, ki jo sprejema, je ugotovljena;
- **posvojitev** (ang. Adoption). Dosežena je odločitev za sprejem inovacije;
- **prilagoditev** (ang. Adaptation). Inovacija je prilagojena in pripravljena za uporabo v organizaciji; člani skupine so usposobljeni za njeno uporabo;
- **uporaba** (ang. Acceptance). Člani skupine se zavežejo k uporabi inovacije;
- **rutinizacija** (ang. Routinization). Uporaba inovacije postane običajna aktivnost;
- **infuzija** (ang. Infusion). Inovacija se uporablja na celovit način, s čimer se učinkovitost skupine povečuje [24].

Možni odgovori predstavljajo vsako od omenjenih stopenj od 1 do 6, pri čemer 1 nakazuje, da posameznik za proces oz. element še ni slišal, in 6 nakazuje, da ga dobro pozna in izvaja rutinsko.

	Povprečje (1–6)	*Mediana*	*Standardni odklon*
Vzdrževanje seznama zahtev	3,71	4	1,38
Uporaba uporabniških zgodb	3,71	4	1,38
Sodelovanje s produktnim vodjo	3,71	4	1,11
Ocenjevanje po metodi Planning Poker	2,57	2	1,40
Ocenjevanje po metodi Team Estimation Game	2,86	4	1,77
Načrtovanje izdaje	3,29	4	1,25
Načrtovanje iteracije	3,14	3	1,35
Vzdrževanje seznama nalog za iteracijo	3,71	4	1,38
Izvajanje Daily Scrum sestankov	4,71	5	1,11
Uporaba pogoriščnih diagramov	3,71	4	1,80
Upoštevanje koncepta "done"	4,14	4	0,90
Izvajanje Sprint Review sestankov	3,71	4	1,25
Izvajanje Sprint Retrospective sestankov	3,71	4	1,50
Poznavanje aktivnosti posameznih vlog	4,00	4	1,29

Tabela 4: Analiza splošnega poznavanja metode Scrum anketirancev

Rezultati kažejo, da najbolje poznamo izvajanje hitrih dnevnih sestankov za pregled narejenih nalog, koncept “done” in vloge v metodi Scrum, kar je smiselno, saj se s temi praksami posamezniki pri izvajanju projektov največkrat srečajo.

Najslabše poznamo obe tehniki ocenjevanja uporabniških zgodb, predvsem zato ker tehnike uporabljamo na dlje trajajočih projektih, s katerimi se ukvarja manjši del razvojne skupine. Zanimivo je, da ima tehnika Planning Poker manjšo sprejetje kot Team Estimation Game, s prvo tehniko se je namreč zagotovo srečala vsaj polovica članov razvojne skupine; mogoče so z njo v praksi seznanjeni, ne poznajo pa uradnega imena. Menimo, da uvedba inovacij ocenjevanja ni ključnega pomena pri uvajanju metodologije, do efektivne ocene količine dela se namreč lahko pride tudi na drugačne načine, ki bolje ustrezajo procesom v podjetju.

Povprečna ocena rezultatov kaže na 4. stopnjo sprejetja inovacije metode Scrum, torej da so se člani skupine zavezali k uporabi inovacije.

5.1.3 Teorija difuzije inovacije

Tretji del vprašalnika se nanaša na Rogersovo teorijo difuzije inovacije [20] (ang. The Diffusion of Inovation theory), katera pojasnjuje, kako se inovacija širi znotraj družbenega sistema v daljšem časovnem obdobju. Končni rezultat difuzije je, da ljudje sprejmejo inovacijo, pa naj bo to ideja, obnašanje ali izdelek. Sprejetje pomeni, da oseba dela nekaj drugače, kot je delala prej. Teorija identificira pet skupin faktorjev (ang. factors), ki vplivajo na dolgoročno sprejemljivost inovacije:

- **faktorji inovacije** (ang. innovation factors),
- **faktorji posameznika** (ang. individual factors),
- **faktorji naloge** (ang. task factors),
- **faktorji okolja** (ang. environmental factors),
- **faktorji organizacije** (ang. organizational factors) [24].

Za vsako od skupin smo posameznikom zastavili vprašanja, predstavljena v preostanku poglavja, na katera so odgovorili s številom od 1 do 7, kjer 1 pomeni, da se s trditvijo sploh ne strinjajo, in 7, da se popolnoma strinjajo. Vprašanja so povzeta po virih [15] in [24]. Namen analize vsake od skupin je, da se ugotovi, kateri faktorji najbolj vplivajo na sprejetje inovacije, torej imajo najvišjo oceno. Od teh je odvisno, ali se inovacija sprejme ali ne.

5.1.3.1 Faktorji inovacije

	Povprečje	*Mediana*	*St. odklon*
Prednost metode Scrum pred drugimi metodami	5,00	6	1,29
Enostavnost uporabe metode Scrum	4,57	5	1,40
Ujemanje metode z osebnimi vrednotami	5,29	6	1,38
Možnost videti, kako druge skupine uporabljajo Scrum	3,43	3	1,99
Moč enostavnega praktičnega preizkusa v delovnem okolju	4,86	5	1,07
Metoda ne povzroča dodatnih stroškov in napora	4,86	5	1,57
Metoda razrešuje veliko težav pri razvoju	4,86	5	1,95
Uporaba metode, ker postaja standard	6,00	6	0,58
Metoda je naprednejša od drugih načinov razvoja	5,00	6	2,00

Tabela 5: Vpliv sprejemljivosti inovacije glede na faktorje inovacije

Najvplivnejši faktor sprejetja metode v našem podjetju, glede na mediano in povprečje, predstavljeno v tabeli 5, je, da metoda Scrum postaja standard. Sledi faktor ujemanja metode Scrum z načini dela oz. osebnimi vrednotami, prednosti metode pred drugimi, prej uporabljenimi metodami razvoja in faktor naprednosti metode od drugih načinov razvoja.

5.1.3.2 Faktorji posameznika

	Povprečje	*Mediana*	*St. odklon*
Možnost samostojnega eksperimentiranja	4,29	5,00	1,38
Priporočilo prijateljev oz. kolegov	3,57	3,00	2,15
Možnost enostavne prilagoditve načinu dela	4,43	6,00	2,37
Možnost priučenja in ovrednotenja med praktičnim delom	5,00	6,00	2,00

Tabela 6: Vpliv sprejemljivosti inovacije glede na faktorje posameznika

Rezultati vprašalnika o vplivih faktorjev posameznika kažejo, da je anketirancem najbolj pomembno, da uporabljajo metodo Scrum, ker se jo lahko naučijo in ovrednotijo med praktičnim delom na projektu razvoja programskih rešitev. Z enako mediano so ovrednotili možnost enostavne prilagoditve metode njihovemu načinu dela.

5.1.3.3 Faktorji o nalogah

	Povprečje	*Mediana*	*St. odklon*
Pomembnost prispevka k večjemu zadovoljstvu naročnika	4,71	5,00	1,38
Skladen je z mojimi potrebami pri razvoju	4,43	5,00	1,72
Pri razvoju olajša izvedbo zahtevnejših nalog	4,14	4,00	1,68

Tabela 7: Vpliv sprejemljivosti inovacije glede na faktorje o nalogah

Rezultati ankete o vplivih faktorjev o nalogah se vsi gibljejo okoli srednjih vrednosti in s tem nakazujejo, da pri nas trenutno ne predstavljajo pomembnega vpliva pri uvedbi metode. Od vseh najbolj izstopa pomembnost prispevka k večjemu zadovoljstvu naročnika s končno programsko rešitvijo.

5.1.3.4 Faktorji okolja

	Povprečje	*Mediana*	*St. odklon*
Metoda je skladna z vrednotami delovnega okolja	5,29	6,00	1,38
Za uporabo imamo na voljo vso potrebno tehnološko infrastrukturo	5,86	6,00	0,69
Metodo uporabljamo, kot je predpisana v literaturi	2,86	2,00	1,77
Razpolaganje z vsemi potrebnimi viri, tj. literatura, izobraževanje, čas za delo, dostop do trenerja Scrum	4,57	5,00	1,51

Tabela 8: Vpliv sprejemljivosti inovacije glede na faktorje okolja

Rezultati vpliva sprejemljivosti inovacije glede na faktorje okolja močno nakazujejo na pomembnost vse potrebne infrastrukture za izvajanje metode, predvsem dobre programske opreme, ki nudi celovito rešitev za vse vloge. Pomembna je tudi skladnost metode z vrednotami delovnega okolja.

Če organizacija ponuja celovito programsko rešitev za vse vloge metode Scrum, je skoraj že težko, da bo posameznik na drugačen način izvajal svoje delo, še posebej če je uporaba orodja nujna pri vsakodnevnih procesih. To je tesno povezano z vrednotami delovnega okolja, v kolikor so s strani vodilnih poudarjene z metodo Scrum, je vpliv sprejemljivosti toliko večji.

5.1.3.5 Faktorji organizacije

	Povprečje	*Mediana*	*St. odklon*
Uporaba zaradi priporočila sodelavcev	4,00	5,00	2,58
Uporaba zaradi razvoja v skupini s sodelavci, s katerimi se tudi socializiram	4,14	4,00	1,57
Uporaba, ker omogoča veliko spontane in neformalne komunikacije med člani razvojne skupine	5,00	5,00	1,53
Uporaba zaradi veliko izkušenj uporabe drugih metodologij razvoja programske opreme	4,14	5,00	2,12
Uporaba, ker razvojni skupini omogoča samoorganizacijsko usmerjeno reševanje težav	5,86	6,00	1,07
Uporaba, ker ovrednoteni posamezniki večinoma zelo priporočajo uporabo	5,00	5,00	1,00
Uporaba, ker se z vsakim novim uporabnikom znatno poveča njena koristnost za organizacijo	5,00	6,00	1,73

Uporaba, ker sicer vse nove tehnologije hitro osvojim	3,71	3,00	1,80
Glavni razlog je zahteva podjetja, da mora razvoj potekati po metodologiji Scrum	5,00	6,00	2,52

Tabela 9: Vpliv sprejemljivosti inovacije glede na faktorje organizacije

Najpomembnejši faktor sprejemljivosti je možnost samostojnega organiziranja dela znotraj razvojne skupine. To lastnost metode Scrum se v podjetju trudimo upoštevati v največji meri, saj smo prepričani, da je ključnega pomena za vsako visoko produktivno razvojno skupino, posledično pa za dobre rezultate vsakega projekta. Pomembna faktorja sta tudi omogočanje veliko spontane in neformalne komunikacije med člani razvojne skupine in povečanje koristnosti z vsakim novim uporabnikom (razvojne skupine), ki sta v veliki meri povezana s prvim. Opaziti je mogoče tudi strinjanje, da je eden od glavnih razlogov uporabe metode zahteva podjetja.

5.1.4 Primerjava s hipotezami Overhageja in Schlaudererja

Zadnji del vprašalnika je namenjen primerjavi hipotez raziskovalcev Overhageja in Schlaudererja, ki sta delala raziskavo o uvedbi metode Scrum leta 2009 v vodilni nemški zavarovalnici [17]. Postavila sta osem hipotez, kako metodologijo Scrum sprejemajo razvijalci, katere temeljijo na treh faktorjih Rogersove teorije o difuziji: relativni prednosti (prve 4 trditve v tabeli 10), kompatibilnosti (5 in 6 trditev v tabeli 10) in kompleksnosti (zadnji 2 trditvi v tabeli 10). Vsaka od hipotez je predstavljena kot trditev v tabeli 10, prvih 6 sta tako potrdila, sedmo nista niti potrdila niti ovrgla in osmo sta potrdila skupaj z zaznavo negativnega vpliva pri sprejemanju.

	Povprečje	*Mediana*	*St. odklon*
Metoda omogoča hitrejši razvoj programske opreme od tradicionalnih načinov	4,86	6,00	1,86
Metoda omogoča razvoj programske opreme, ki bolj ustreza zahtevam naročnika za razliko od tradicionalnih načinov	5,57	6,00	1,27
Metoda omogoča, da se pri razvoju programske rešitve naučimo več kot pri tradicionalnih načinih	4,86	6,00	1,68
Metoda omogoča večje zadovoljstvo razvijalcev z razvito programsko rešitvijo od tradicionalnih načinov	486	5,00	1,35
Metoda omogoča boljši pregled nad potekom razvoja programske rešitve od tradicionalnih načinov	5,57	5,00	0,79
Metoda omogoča boljše sodelovanje razvijalcev pri razvoju programske rešitve od tradicionalnih načinov	5,43	6,00	1,27
Metoda zmanjša kompleksnost procesa razvoja programske rešitve v primerjavi s tradicionalnimi načini	4,86	5,00	1,68

Metoda zahteva več discipline pri razvoju programske rešitve od tradicionalnih načinov	5,71	6,00	1,11

Tabela 10: Preverjanje hipotez Overhageja in Schlaudereja

Rezultati vprašalnika kažejo na povprečno strinjanje (mediana 6) s trditvami in v treh primerih delno strinjanje s trditvami (mediana 5). Povprečje rezultatov nakazuje na pozitivno strinjanje s trditvami metode, predvsem pri zahtevanju več discipline, omogočanju razvoja programske opreme, ki bolj ustreza zahtevam naročnika, in da ima metoda boljši pregled nad potekom razvoja projekta. Pozitivni rezultati nam nakazujejo, da so posamezniki seznanjeni s prednostmi metodologije, kar dobro vpliva na motivacijo pri uvajanju metode v podjetje.

Če primerjamo rezultate našega vprašalnika s hipotezami Overhageja in Schlaudererja, vse hipoteze razen četrte kažejo enake rezultate, vendar je v nekaterih primerih težje trditi, če smo hipotezo dejansko potrdili ali ne, ker so rezultati relativno blizu srednji vrednosti. Četrte hipoteze, tako kot sedme, ne moramo niti potrditi niti ovreči, ker se večina anketirancev z njo samo delno strinja. Prva in tretja hipoteza imata enako povprečno vrednost kot četrta, vendar sta vseeno sprejeti, ker imata mediano 6 in ne 5, kar nakazuje na večje strinjanje anketirancev.

5.2 Sklepne ugotovitve

Rezultati vprašalnika nakazujejo na delno uvedbo metode v podjetje. Večina anketirancev se strinja, da je metoda Scrum boljša za razvoj spletnih strani in aplikacij kot tradicionalne metode, vendar hkrati določenih elementov metode ne poznajo najbolje. V času izvajanja ankete sta bila v podjetju zaposlena dva nova razvijalca, ki metode Scrum še nista poznala, kar je glede na število zaposlenih v podjetju relativno močno vplivalo na rezultate. Metodo Scrum uporabljamo predvsem pri razvoju spletnih aplikacij in večjih spletnih strani, kjer razvoj traja dlje časa, medtem ko za manjše spletne strani uporabimo tudi tradicionalen pristop.

Zahvaljujoč vprašalniku vemo, do kakšne mere smo metodo Scrum do tega trenutka uvedli v podjetje, izvedeli smo tudi, kateri elementi metode so slabše uvedeni, na kaj se moramo bolj osredotočiti pri uvajanju elementov ter da jih moramo čim prej izboljšati za potencialno doseganje boljših rezultatov. Naslednji cilj uvajanja metode je dvig sprejetja inovacij na povprečno vsaj 5. stopnjo, kar bomo dosegli z boljšim izobraževanjem kadra o sami metodi in z večjo pozornostjo na poudarku sprejetja in razumevanja posameznih uvedenih procesov.

Poglavje 6 Zaključek

Z izvedeno anketo smo analizirali stopnjo uvedbe metode Scrum, katero podjetje postopoma in po potrebi uvaja v svoje delovne procese. Rezultati kažejo na delno uvedbo, torej da so se člani skupine zavezali k uporabi in da puščajo še kar nekaj prostora za dokončno uvedbo ter uporabo metode na celovit način. Izpostavljena odstopanja od teorije v praktičnem primeru uvajanja metode kažejo na to, da glede na trenutne potrebe podjetja vsi procesi metode verjetno ne bodo v celoti sprejeti. Izhodišče za nadaljnje uvajanje metode bodo (ponovne) predstavitve metode vsem njenim uporabnikom, saj se je izkazalo, da je poznavanje določenih osnovnih praks, katere podjetje uporablja pri razvoju, na prenizkem nivoju.

V praktičnem primeru uvajanja smo opazili, da so največja odstopanja od teorije metode na samem začetku, pri planiranju. Podjetje avtorja diplomskega dela si težko privošči netočne ocene trajanja projekta v obliki točk, saj je treba ocene upoštevati že pri izdelavi ponudbe projekta, ocena pa ne sme biti previsoka zaradi konkurenčnosti na trgu in ne prenizka zaradi tveganja deficita. Taka, vnaprej planirana ponudba, že vsebuje na sklope razdeljeno delo, ocenjeno v urah, in je zato v primeru manjših projektov smiselno, da se sklopi iz ponudbe uporabijo kot neke vrste uporabniške zgodbe skupaj z njihovimi ocenami v urah. Časovno je za podjetje tak postopek bolj optimalen in bolj spominja na metodo Waterfall kot Scrum. V primeru večjih oz. časovno dlje trajajočih projektov se je metoda Scrum v našem podjetju izkazala za primernejšo. Glede na preteklo uspešnost in trenutno stopnjo uvedbe je priporočljivo, da se sprejetje in sami procesi metode dvignejo na višjo raven, kot je uvedena sedaj. Raziskati je treba, ali obstaja celovita rešitev, ki bo omogočala enak delovni proces v primeru tako velikih kot manjših projektov. Taka rešitev bo pomagala tudi pri poenostavljanju delovnih procesov zaposlenih v podjetju, saj bodo na vseh projektih izvajali enake procese, za kar smo prepričani, da bo dobro vplivalo na njihovo efektivnost in boljše rezultate podjetja.

Literatura

[1] (2012) Agilemanifesto. Dostopno na: http://www.agilemanifesto.org/iso/sl/ (dostop 11.6.2016).

[2] AngularJS. Dostopno na: https://en.wikipedia.org/wiki/AngularJS (dostop 28.6.2016).

[3] Bamboo (software). Dostopno na: https://en.wikipedia.org/wiki/Bamboo_(software) (dostop 28.6.2016).

[4] Bitbucket. Dostopno na: https://en.wikipedia.org/wiki/Bitbucket (dostop 29.6.2016).

[5] ClydeBank Business, Agile Project Management: QuickStart Guide, ClydeBank Media LLC, 2014.

[6] Mike Cohn, Succeeding with Agile: Software Development Using Scrum 1st Edition, Addison-Wesley Professional, 2009.

[7] Mike Cohn, User Stories Applied For Agile Software Development, Addison-Wesley Professional, 2004.

[8] R. B. Cooper, R. W. Zmud, "Information technology implementation research: A technological diffusion approach", Management Science, št. 36, zv. 2, str. 123–139, 1990.

[9] (2007) EMRIS. Dostopno na: http://www2.gov.si/mju/emris.nsf/0/4F5A54F79681D61AC1256E9E003E08C8 (dostop 15.6.2016).

[10] GitHub. Dostopno na: https://en.wikipedia.org/wiki/GitHub (dostop 29.6.2016).

[11] Jira (software). Dostopno na: https://en.wikipedia.org/wiki/Jira_(software) (dostop 19.6.2016).

[12] Kanban board. Dostopno na: https://en.wikipedia.org/wiki/Kanban_board (dostop 23.6.2016).

[13] T. H. Kwon, R. W. Zmud, Unifying the fragmented models of information systems implementation, v R. J. Boland, R. A. Hirschheim: Critical Issues in Information Systems Research, New York: Wiley & Sons, 1987, str. 227–251.

[14] Viljan Mahnič, Problemi in rešitve pri uvajanju metode Scrum v proces razvoja programske opreme, Zbornik 18. konference "Dnevi slovenske informatike", Portorož, april 2011.

[15] E. Mustonen-Ollila and K. Lyytinen, "Why Organizations Adopt Information Systems Process Innovations: A Longitudinal Study using Diffusion of Innovation Theory," Information Systems Journal, 2003, pp. 275-297.

[16] Node.js. Dostopno na: https://en.wikipedia.org/wiki/Node.js (dostop 29.6.2016).

[17] S. Overhage, S. Schlauderer, "Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects", v zborniku 45th Hawaii International Conference on System Sciences, Maui, Hawaii, jan. 2012, str. 5452–5461.

[18] Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK® Guide) Fifth Edition. Pennsylvania, 2012.

[19] ReactJS.NET. Dostopno na: https://en.wikipedia.org/wiki/React_(JavaScript_library) (dostop 29.6.2016).

[20] Everett M. Rogers, Diffusion of Innovations, 4th Edition, Free Press, 1995.

[21] Ken Schwaber, Agile Project Management with Scrum, Microsoft Press - Microsoft Corporation, 2004.

[22] (2008) Scrum Alliance. Dostopno na: https://www.scrumalliance.org/community/articles/2008/september/definition-of-done-a-reference (dostop 20.6.2016).

[23] (2012) Scrum Breakfast. Dostopno na: http://www.scrum-breakfast.com/2012/11/sample-definition-of-done.html (dostop 9.6.2016).

[24] Neža Štrukelj, Uvajanje metodologije Scrum na večjem projektu: študija primera in analiza faktorjev sprejemljivosti (magistrsko delo), Ljubljana, Fak. za računalništvo in informatiko, 2016, str. 27-32.

[25] Waterfall model. Dostopno na: https://en.wikipedia.org/wiki/Waterfall_model (dostop 27.6.2016).

[26] Laurie Williams, Agile software development methodologies and practices, Advances in Computers, 80, 2010, pp. 1-4